

general\_ledger

Generated by Doxygen 1.8.1.2

Sun Jun 15 2014 18:09:13



# Contents

<b>1</b>	<b>General Ledger.</b>	<b>1</b>
<b>2</b>	<b>Todo List</b>	<b>3</b>
<b>3</b>	<b>Module Index</b>	<b>5</b>
3.1	Modules . . . . .	5
<b>4</b>	<b>Class Index</b>	<b>7</b>
4.1	Class Hierarchy . . . . .	7
<b>5</b>	<b>Class Index</b>	<b>9</b>
5.1	Class List . . . . .	9
<b>6</b>	<b>File Index</b>	<b>11</b>
6.1	File List . . . . .	11
<b>7</b>	<b>Module Documentation</b>	<b>13</b>
7.1	General Ledger database module. . . . .	13
7.1.1	Detailed Description . . . . .	13
7.2	Database interaction module . . . . .	14
7.2.1	Detailed Description . . . . .	14
7.2.2	Function Documentation . . . . .	15
7.2.2.1	get_connection . . . . .	15
7.2.2.2	get_database_type . . . . .	15
7.3	SQL statements module . . . . .	16
7.3.1	Detailed Description . . . . .	16
7.4	Program configuration module . . . . .	17
7.4.1	Detailed Description . . . . .	17
7.5	General purpose helpers. . . . .	18
7.5.1	Detailed Description . . . . .	18
7.5.2	Function Documentation . . . . .	18
7.5.2.1	split . . . . .	18
7.5.2.2	split . . . . .	18
7.5.2.3	trim . . . . .	18

7.5.2.4	trim_back	19
7.5.2.5	trim_front	19
7.6	Reporting program.	20
7.6.1	Detailed Description	20
7.6.2	Function Documentation	20
7.6.2.1	login	20
7.6.2.2	main	20
7.6.2.3	set_configuration	21
7.7	Database program.	22
7.7.1	Detailed Description	22
7.7.2	Function Documentation	22
7.7.2.1	check_db_parameters	22
7.7.2.2	check_help_and_version	22
7.7.2.3	login	23
7.7.2.4	main	23
7.7.2.5	set_configuration	23
<b>8</b>	<b>Class Documentation</b>	<b>25</b>
8.1	genleg::Config Class Reference	25
8.1.1	Detailed Description	25
8.1.2	Constructor & Destructor Documentation	25
8.1.2.1	Config	25
8.1.2.2	~Config	26
8.1.3	Member Function Documentation	26
8.1.3.1	add_cmdline_option	26
8.1.3.2	is_set	26
8.1.3.3	operator[]	26
8.1.3.4	populate_from_cmdline	26
8.1.3.5	populate_from_file	27
8.1.4	Member Data Documentation	27
8.1.4.1	m_opts_set	27
8.1.4.2	m_opts_supp	27
8.2	genleg::ConfigBadConfigFile Class Reference	27
8.2.1	Detailed Description	28
8.2.2	Constructor & Destructor Documentation	28
8.2.2.1	ConfigBadConfigFile	28
8.3	genleg::ConfigBadOption Class Reference	29
8.3.1	Detailed Description	29
8.3.2	Constructor & Destructor Documentation	29
8.3.2.1	ConfigBadOption	30

8.4	<a href="#">genleg::ConfigCouldNotOpenFile Class Reference</a>	30
8.4.1	<a href="#">Detailed Description</a>	31
8.4.2	<a href="#">Constructor &amp; Destructor Documentation</a>	31
8.4.2.1	<a href="#">ConfigCouldNotOpenFile</a>	31
8.5	<a href="#">genleg::ConfigException Class Reference</a>	31
8.5.1	<a href="#">Detailed Description</a>	31
8.5.2	<a href="#">Constructor &amp; Destructor Documentation</a>	32
8.5.2.1	<a href="#">ConfigException</a>	32
8.6	<a href="#">genleg::ConfigOptionNotSet Class Reference</a>	32
8.6.1	<a href="#">Detailed Description</a>	33
8.6.2	<a href="#">Constructor &amp; Destructor Documentation</a>	33
8.6.2.1	<a href="#">ConfigOptionNotSet</a>	33
8.7	<a href="#">gldb::DBConn Class Reference</a>	33
8.7.1	<a href="#">Detailed Description</a>	34
8.7.2	<a href="#">Constructor &amp; Destructor Documentation</a>	34
8.7.2.1	<a href="#">DBConn</a>	34
8.7.2.2	<a href="#">DBConn</a>	34
8.7.3	<a href="#">Member Function Documentation</a>	34
8.7.3.1	<a href="#">operator=</a>	34
8.7.3.2	<a href="#">query</a>	34
8.7.3.3	<a href="#">select</a>	34
8.7.4	<a href="#">Member Data Documentation</a>	35
8.7.4.1	<a href="#">m_imp</a>	35
8.8	<a href="#">gldb::DBConnCouldNotConnect Class Reference</a>	35
8.8.1	<a href="#">Detailed Description</a>	36
8.8.2	<a href="#">Constructor &amp; Destructor Documentation</a>	36
8.8.2.1	<a href="#">DBConnCouldNotConnect</a>	36
8.9	<a href="#">gldb::DBConnCouldNotQuery Class Reference</a>	36
8.9.1	<a href="#">Detailed Description</a>	37
8.9.2	<a href="#">Constructor &amp; Destructor Documentation</a>	37
8.9.2.1	<a href="#">DBConnCouldNotQuery</a>	37
8.10	<a href="#">gldb::DBConnDummy Class Reference</a>	37
8.10.1	<a href="#">Detailed Description</a>	38
8.10.2	<a href="#">Constructor &amp; Destructor Documentation</a>	38
8.10.2.1	<a href="#">DBConnDummy</a>	38
8.10.2.2	<a href="#">DBConnDummy</a>	39
8.10.2.3	<a href="#">~DBConnDummy</a>	39
8.10.3	<a href="#">Member Function Documentation</a>	39
8.10.3.1	<a href="#">operator=</a>	39
8.10.3.2	<a href="#">select</a>	39

8.11	<a href="#">gldb::DBConnException Class Reference</a>	39
8.11.1	<a href="#">Detailed Description</a>	40
8.11.2	<a href="#">Constructor &amp; Destructor Documentation</a>	40
8.11.2.1	<a href="#">DBConnException</a>	40
8.12	<a href="#">gldb::DBConnImp Class Reference</a>	40
8.12.1	<a href="#">Detailed Description</a>	41
8.12.2	<a href="#">Constructor &amp; Destructor Documentation</a>	41
8.12.2.1	<a href="#">DBConnImp</a>	41
8.12.2.2	<a href="#">~DBConnImp</a>	41
8.12.3	<a href="#">Member Function Documentation</a>	41
8.12.3.1	<a href="#">query</a>	41
8.12.3.2	<a href="#">select</a>	42
8.13	<a href="#">gldb::DBConnMySQL Class Reference</a>	42
8.13.1	<a href="#">Detailed Description</a>	43
8.13.2	<a href="#">Constructor &amp; Destructor Documentation</a>	43
8.13.2.1	<a href="#">DBConnMySQL</a>	43
8.13.2.2	<a href="#">DBConnMySQL</a>	43
8.13.2.3	<a href="#">~DBConnMySQL</a>	43
8.13.3	<a href="#">Member Function Documentation</a>	43
8.13.3.1	<a href="#">operator=</a>	44
8.13.3.2	<a href="#">query</a>	44
8.13.3.3	<a href="#">select</a>	44
8.13.4	<a href="#">Member Data Documentation</a>	44
8.13.4.1	<a href="#">m_conn</a>	44
8.14	<a href="#">genleg::DBSQLMySQL Class Reference</a>	44
8.14.1	<a href="#">Detailed Description</a>	45
8.15	<a href="#">genleg::DBSQLStatements Class Reference</a>	45
8.15.1	<a href="#">Detailed Description</a>	46
8.15.2	<a href="#">Constructor &amp; Destructor Documentation</a>	46
8.15.2.1	<a href="#">DBSQLStatements</a>	46
8.15.2.2	<a href="#">~DBSQLStatements</a>	47
8.15.3	<a href="#">Member Function Documentation</a>	47
8.15.3.1	<a href="#">create_table</a>	47
8.15.3.2	<a href="#">create_view</a>	47
8.15.3.3	<a href="#">drop_table</a>	47
8.15.3.4	<a href="#">drop_view</a>	47
8.15.3.5	<a href="#">get_perms</a>	48
8.15.3.6	<a href="#">grant</a>	48
8.15.3.7	<a href="#">revoke</a>	48
8.15.3.8	<a href="#">update_user</a>	48

8.15.3.9	user_by_id	49
8.15.3.10	user_by_username	49
8.16	genleg::GLDatabase Class Reference	49
8.16.1	Detailed Description	51
8.16.2	Constructor & Destructor Documentation	51
8.16.2.1	GLDatabase	51
8.16.2.2	~GLDatabase	51
8.16.3	Member Function Documentation	51
8.16.3.1	backend	51
8.16.3.2	create_structure	52
8.16.3.3	create_user	52
8.16.3.4	destroy_structure	52
8.16.3.5	get_user_by_id	52
8.16.3.6	get_user_by_username	52
8.16.3.7	grant	53
8.16.3.8	load_sample_data	53
8.16.3.9	revoke	53
8.16.3.10	update_user	53
8.16.4	Member Data Documentation	53
8.16.4.1	m_dbc	54
8.16.4.2	m_sql	54
8.16.4.3	m_tables	54
8.16.4.4	m_views	54
8.17	genleg::GLDBException Class Reference	54
8.17.1	Detailed Description	54
8.17.2	Constructor & Destructor Documentation	54
8.17.2.1	GLDBException	54
8.18	genleg::GLUser Class Reference	55
8.18.1	Detailed Description	56
8.18.2	Constructor & Destructor Documentation	56
8.18.2.1	GLUser	56
8.18.2.2	~GLUser	56
8.18.3	Member Function Documentation	56
8.18.3.1	check_password	57
8.18.3.2	enabled	57
8.18.3.3	firstname	57
8.18.3.4	id	57
8.18.3.5	lastname	57
8.18.3.6	pass_hash	57
8.18.3.7	pass_salt	58

8.18.3.8	permissions	58
8.18.3.9	set_enabled	58
8.18.3.10	set_firstname	58
8.18.3.11	set_lastname	58
8.18.3.12	set_password	58
8.18.3.13	set_username	58
8.18.3.14	username	59
8.18.4	Member Data Documentation	59
8.18.4.1	m_enabled	59
8.18.4.2	m_firstname	59
8.18.4.3	m_id	59
8.18.4.4	m_lastname	59
8.18.4.5	m_pass_hash	59
8.18.4.6	m_pass_salt	59
8.18.4.7	m_perms	59
8.18.4.8	m_username	59
8.19	gldb::Table Class Reference	60
8.19.1	Detailed Description	61
8.19.2	Constructor & Destructor Documentation	61
8.19.2.1	Table	61
8.19.2.2	~Table	61
8.19.3	Member Function Documentation	61
8.19.3.1	append_record	61
8.19.3.2	create_from_file	61
8.19.3.3	get_field	62
8.19.3.4	get_headers	62
8.19.3.5	insert_query	62
8.19.3.6	num_fields	62
8.19.3.7	num_records	62
8.19.3.8	operator[]	63
8.19.3.9	set_quoted	63
8.19.4	Member Data Documentation	63
8.19.4.1	m_headers	63
8.19.4.2	m_quoted	63
8.19.4.3	m_records	63
8.20	gldb::TableBadInputFile Class Reference	63
8.20.1	Detailed Description	64
8.20.2	Constructor & Destructor Documentation	64
8.20.2.1	TableBadInputFile	64
8.21	gldb::TableCouldNotOpenInputFile Class Reference	65



8.21.1 Detailed Description . . . . .	65
8.21.2 Constructor & Destructor Documentation . . . . .	65
8.21.2.1 TableCouldNotOpenInputFile . . . . .	66
8.22 glDb::TableException Class Reference . . . . .	66
8.22.1 Detailed Description . . . . .	66
8.22.2 Constructor & Destructor Documentation . . . . .	66
8.22.2.1 TableException . . . . .	67
8.23 glDb::TableField Class Reference . . . . .	67
8.23.1 Detailed Description . . . . .	68
8.23.2 Constructor & Destructor Documentation . . . . .	68
8.23.2.1 TableField . . . . .	68
8.23.2.2 TableField . . . . .	68
8.23.2.3 ~TableField . . . . .	68
8.23.3 Member Function Documentation . . . . .	68
8.23.3.1 length . . . . .	68
8.23.3.2 operator std::string . . . . .	69
8.23.3.3 operator+= . . . . .	69
8.23.3.4 operator+= . . . . .	69
8.23.3.5 operator= . . . . .	69
8.23.3.6 operator= . . . . .	69
8.23.3.7 operator[] . . . . .	70
8.23.3.8 operator[] . . . . .	70
8.23.4 Friends And Related Function Documentation . . . . .	70
8.23.4.1 operator<< . . . . .	70
8.23.5 Member Data Documentation . . . . .	70
8.23.5.1 m_data . . . . .	70
8.24 glDb::TableMismatchedRecordLength Class Reference . . . . .	70
8.24.1 Detailed Description . . . . .	71
8.24.2 Constructor & Destructor Documentation . . . . .	71
8.24.2.1 TableMismatchedRecordLength . . . . .	71
8.25 glDb::TableNoSuchField Class Reference . . . . .	72
8.25.1 Detailed Description . . . . .	72
8.25.2 Constructor & Destructor Documentation . . . . .	73
8.25.2.1 TableNoSuchField . . . . .	73
8.26 glDb::TableNoSuchRecord Class Reference . . . . .	73
8.26.1 Detailed Description . . . . .	74
8.26.2 Constructor & Destructor Documentation . . . . .	74
8.26.2.1 TableNoSuchRecord . . . . .	74
8.27 glDb::TableRow Class Reference . . . . .	74
8.27.1 Detailed Description . . . . .	75

8.27.2	Constructor & Destructor Documentation	75
8.27.2.1	TableRow	75
8.27.2.2	TableRow	75
8.27.2.3	TableRow	75
8.27.2.4	~TableRow	75
8.27.3	Member Function Documentation	75
8.27.3.1	append_field	75
8.27.3.2	append_field	75
8.27.3.3	append_field	76
8.27.3.4	operator[]	76
8.27.3.5	operator[]	76
8.27.3.6	print	76
8.27.3.7	record_string	76
8.27.3.8	record_string	77
8.27.3.9	size	77
8.27.4	Member Data Documentation	77
8.27.4.1	m_fields	77
<b>9</b>	<b>File Documentation</b>	<b>79</b>
9.1	lib/config/config.cpp File Reference	79
9.1.1	Detailed Description	79
9.2	lib/config/config.h File Reference	80
9.2.1	Detailed Description	81
9.3	lib/config/config_getopt.cpp File Reference	81
9.3.1	Detailed Description	81
9.3.2	Macro Definition Documentation	82
9.3.2.1	_XOPEN_SOURCE	82
9.4	lib/database/data_structures.h File Reference	82
9.4.1	Detailed Description	83
9.5	lib/database/database.h File Reference	83
9.5.1	Detailed Description	85
9.6	lib/database/dbconn.cpp File Reference	85
9.6.1	Detailed Description	85
9.7	lib/database/dbconn.h File Reference	86
9.7.1	Detailed Description	87
9.8	lib/database/dbconnimp.h File Reference	87
9.8.1	Detailed Description	89
9.9	lib/database/table.cpp File Reference	89
9.9.1	Detailed Description	89
9.10	lib/database/table.h File Reference	90

9.10.1 Detailed Description . . . . .	91
9.11 lib/database/tablefield.cpp File Reference . . . . .	92
9.11.1 Detailed Description . . . . .	92
9.12 lib/database/tablefield.h File Reference . . . . .	92
9.12.1 Detailed Description . . . . .	94
9.13 lib/database/ablerow.cpp File Reference . . . . .	94
9.13.1 Detailed Description . . . . .	94
9.14 lib/database/ablerow.h File Reference . . . . .	95
9.14.1 Detailed Description . . . . .	96
9.15 lib/database_imp/database_imp.h File Reference . . . . .	96
9.15.1 Detailed Description . . . . .	98
9.16 lib/database_imp/dummy/dbconn_dummy_imp.cpp File Reference . . . . .	98
9.16.1 Detailed Description . . . . .	99
9.17 lib/database_imp/dummy/dbconn_dummy_imp.h File Reference . . . . .	99
9.17.1 Detailed Description . . . . .	101
9.18 lib/database_imp/mysql/dbconn_mysql_imp.cpp File Reference . . . . .	101
9.18.1 Detailed Description . . . . .	102
9.19 lib/database_imp/mysql/dbconn_mysql_imp.h File Reference . . . . .	102
9.19.1 Detailed Description . . . . .	104
9.20 lib/dbsql/dbsql.h File Reference . . . . .	104
9.20.1 Detailed Description . . . . .	105
9.21 lib/dbsql/dbsql_implementations.h File Reference . . . . .	105
9.21.1 Detailed Description . . . . .	106
9.22 lib/dbsql/dbsql_mysql.h File Reference . . . . .	107
9.22.1 Detailed Description . . . . .	108
9.23 lib/dbsql/dbsqlstatements.cpp File Reference . . . . .	108
9.23.1 Detailed Description . . . . .	108
9.24 lib/dbsql/dbsqlstatements.h File Reference . . . . .	109
9.24.1 Detailed Description . . . . .	110
9.25 lib/gldb/gldatabase.cpp File Reference . . . . .	110
9.25.1 Detailed Description . . . . .	111
9.25.2 Function Documentation . . . . .	111
9.25.2.1 boolstring_to_bool . . . . .	111
9.26 lib/gldb/gldatabase.h File Reference . . . . .	111
9.26.1 Detailed Description . . . . .	113
9.27 lib/gldb/gldb.h File Reference . . . . .	113
9.27.1 Detailed Description . . . . .	114
9.28 lib/gldb/glexception.h File Reference . . . . .	114
9.28.1 Detailed Description . . . . .	115
9.29 lib/gldb/gluser.cpp File Reference . . . . .	116

9.29.1 Detailed Description . . . . .	116
9.30 lib/gldb/gluser.h File Reference . . . . .	116
9.30.1 Detailed Description . . . . .	117
9.31 lib/gldb/gluser_pass.cpp File Reference . . . . .	118
9.31.1 Detailed Description . . . . .	118
9.31.2 Macro Definition Documentation . . . . .	119
9.31.2.1 _XOPEN_SOURCE . . . . .	119
9.31.3 Function Documentation . . . . .	119
9.31.3.1 generate_salt . . . . .	119
9.32 lib/stringhelp/stringhelp.cpp File Reference . . . . .	119
9.32.1 Detailed Description . . . . .	119
9.33 lib/stringhelp/stringhelp.h File Reference . . . . .	120
9.33.1 Detailed Description . . . . .	121
9.34 progs/gl_db/gl_db_main.cpp File Reference . . . . .	121
9.34.1 Detailed Description . . . . .	122
9.35 progs/gl_report/gl_report_main.cpp File Reference . . . . .	122
9.35.1 Detailed Description . . . . .	124
9.36 progs/gl_user/gl_user_main.cpp File Reference . . . . .	124
9.36.1 Detailed Description . . . . .	125
9.36.2 Function Documentation . . . . .	125
9.36.2.1 check_db_parameters . . . . .	125
9.36.2.2 check_help_and_version . . . . .	126
9.36.2.3 check_user_password . . . . .	126
9.36.2.4 enable_user . . . . .	126
9.36.2.5 get_user . . . . .	126
9.36.2.6 login . . . . .	126
9.36.2.7 main . . . . .	127
9.36.2.8 set_configuration . . . . .	127
9.36.2.9 set_user_password . . . . .	127
9.36.2.10 show_user_details . . . . .	127

## 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

## Todo List

File [gluser\\_pass.cpp](#)

Implement a better form of password encryption. In particular, these functions are not re-entrant, and only use the first 8 characters of the password.





## Chapter 3

# Module Index

### 3.1 Modules

Here is a list of all modules:

General Ledger database module. . . . .	13
Database interaction module . . . . .	14
SQL statements module . . . . .	16
Program configuration module . . . . .	17
General purpose helpers. . . . .	18
Reporting program. . . . .	20
Database program. . . . .	22



## Chapter 4

# Class Index

### 4.1 Class Hierarchy

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

genleg::Config . . . . .	25
genleg::ConfigException . . . . .	31
genleg::ConfigBadConfigFile . . . . .	27
genleg::ConfigBadOption . . . . .	29
genleg::ConfigCouldNotOpenFile . . . . .	30
genleg::ConfigOptionNotSet . . . . .	32
gldb::DBConn . . . . .	33
gldb::DBConnException . . . . .	39
gldb::DBConnCouldNotConnect . . . . .	35
gldb::DBConnCouldNotQuery . . . . .	36
gldb::DBConnImp . . . . .	40
gldb::DBConnDummy . . . . .	37
gldb::DBConnMySQL . . . . .	42
genleg::DBSQLStatements . . . . .	45
genleg::DBSQLMySQL . . . . .	44
genleg::GLDatabase . . . . .	49
genleg::GLDBException . . . . .	54
genleg::GLUser . . . . .	55
gldb::Table . . . . .	60
gldb::TableException . . . . .	66
gldb::TableBadInputFile . . . . .	63
gldb::TableCouldNotOpenInputFile . . . . .	65
gldb::TableMismatchedRecordLength . . . . .	70
gldb::TableNoSuchField . . . . .	72
gldb::TableNoSuchRecord . . . . .	73
gldb::TableField . . . . .	67
gldb::TableRow . . . . .	74



## Chapter 5

# Class Index

### 5.1 Class List

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

<a href="#">genleg::Config</a>	Configuration options class . . . . .	25
<a href="#">genleg::ConfigBadConfigFile</a>	Exception class for badly formed configuration file . . . . .	27
<a href="#">genleg::ConfigBadOption</a>	Exception class for bad provided option . . . . .	29
<a href="#">genleg::ConfigCouldNotOpenFile</a>	Exception class for when conf file cannot be opened . . . . .	30
<a href="#">genleg::ConfigException</a>	Configuration module exception base class . . . . .	31
<a href="#">genleg::ConfigOptionNotSet</a>	Exception class for option not set . . . . .	32
<a href="#">gldb::DBConn</a>	Database connection class . . . . .	33
<a href="#">gldb::DBConnCouldNotConnect</a>	Could not connect to database exception class . . . . .	35
<a href="#">gldb::DBConnCouldNotQuery</a>	Could not execute database query exception class . . . . .	36
<a href="#">gldb::DBConnDummy</a>	Dummy database implementation class . . . . .	37
<a href="#">gldb::DBConnException</a>	Base database connection exception class . . . . .	39
<a href="#">gldb::DBConnImp</a>	Abstract database implementation base class . . . . .	40
<a href="#">gldb::DBConnMySQL</a>	MySQL database implementation class . . . . .	42
<a href="#">genleg::DBSQLMySQL</a>	MySQL SQL statements class . . . . .	44
<a href="#">genleg::DBSQLStatements</a>	SQL statements class . . . . .	45
<a href="#">genleg::GLDatabase</a>	General ledger database class . . . . .	49
<a href="#">genleg::GLDBException</a>	Base general ledger database exceptionc class . . . . .	54
<a href="#">genleg::GLUser</a>	General ledger user class . . . . .	55
<a href="#">gldb::Table</a>	Database table class . . . . .	60

<a href="#">gldb::TableBadInputFile</a>	
Could not connect to database exception class . . . . .	63
<a href="#">gldb::TableCouldNotOpenInputFile</a>	
Could not connect to database exception class . . . . .	65
<a href="#">gldb::TableException</a>	
Base database connection exception class . . . . .	66
<a href="#">gldb::TableField</a>	
Database table field class . . . . .	67
<a href="#">gldb::TableMismatchedRecordLength</a>	
Mismatched record length exception class . . . . .	70
<a href="#">gldb::TableNoSuchField</a>	
No such field exception class . . . . .	72
<a href="#">gldb::TableNoSuchRecord</a>	
No such record exception class . . . . .	73
<a href="#">gldb::TableRow</a>	
Database table row class . . . . .	74

## Chapter 6

# File Index

### 6.1 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 . . . . .	79
lib/config/ <a href="#">config.h</a>	
Interface to program configurations class . . . . .	80
lib/config/ <a href="#">config_getopt.cpp</a>	
Implementation of command line functionality . . . . .	81
lib/database/ <a href="#">data_structures.h</a>	
Main interface to database data structures . . . . .	82
lib/database/ <a href="#">database.h</a>	
User interface to database functionality . . . . .	83
lib/database/ <a href="#">dbconn.cpp</a>	
Implementation of database connection class . . . . .	85
lib/database/ <a href="#">dbconn.h</a>	
Interface to database connection base class . . . . .	86
lib/database/ <a href="#">dbconnimp.h</a>	
Interface to abstract database implementation base class . . . . .	87
lib/database/ <a href="#">table.cpp</a>	
Implementation of database table data structure . . . . .	89
lib/database/ <a href="#">table.h</a>	
Interface to database table data structure . . . . .	90
lib/database/ <a href="#">tablefield.cpp</a>	
Implementation of database table field class . . . . .	92
lib/database/ <a href="#">tablefield.h</a>	
Interface to database table field class . . . . .	92
lib/database/ <a href="#">tablerow.cpp</a>	
Implementation of database table row data structure . . . . .	94
lib/database/ <a href="#">tablerow.h</a>	
Interface to database table row data structure . . . . .	95
lib/database_imp/ <a href="#">database_imp.h</a>	
Interface to database implementation factory function . . . . .	96
lib/database_imp/dummy/ <a href="#">dbconn_dummy_imp.cpp</a>	
Implementation of Dummy database connection implementation class . . . . .	98
lib/database_imp/dummy/ <a href="#">dbconn_dummy_imp.h</a>	
Interface to dummy database connection implementation class . . . . .	99
lib/database_imp/mysql/ <a href="#">dbconn_mysql_imp.cpp</a>	
Implementation of MySQL database connection implementation class . . . . .	101
lib/database_imp/mysql/ <a href="#">dbconn_mysql_imp.h</a>	
Interface to MySQL database connection implementation class . . . . .	102

lib/dbsql/dbsql.h	
User interface to DBSQL module	104
lib/dbsql/dbsql_functions.h	??
lib/dbsql/dbsql_implementations.h	
Aggregation header for DBSqlStatements implementations	105
lib/dbsql/dbsql_mysql.h	
Interface to MySQL SQL statement class	107
lib/dbsql/dbsqlstatements.cpp	
Implementation of SQL statement class	108
lib/dbsql/dbsqlstatements.h	
Implementation of SQL module standalone functions	109
lib/gldb/gldatabase.cpp	
Implementation of General Ledger database class	110
lib/gldb/gldatabase.h	
Interface to General Ledger database class	111
lib/gldb/gldb.h	
User interface to General Ledger database module	113
lib/gldb/glexception.h	
Interface to General Ledger base exception class	114
lib/gldb/gluser.cpp	
Implementation of user class	116
lib/gldb/gluser.h	
Interface to user class	116
lib/gldb/gluser_pass.cpp	
Implementation of password functions for user class	118
lib/stringhelp/stringhelp.cpp	
Implementation of string helper functions	119
lib/stringhelp/stringhelp.h	
Interface to string helper functions	120
progs/gl_db/gl_db_main.cpp	
Main functionality for gl_db program	121
progs/gl_report/gl_report_main.cpp	
Main functionality for gl_report program	122
progs/gl_user/gl_user_main.cpp	
Main functionality for gl_user program	124



# Chapter 7

## Module Documentation

### 7.1 General Ledger database module.

#### Classes

- class [genleg::GLDatabase](#)  
*General ledger database class.*
- class [genleg::GLDBException](#)  
*Base general ledger database exceptionc class.*
- class [genleg::GLUser](#)  
*General ledger user class.*

#### 7.1.1 Detailed Description

Module for interacting with the general ledger database model.

## 7.2 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::TableNoSuchField](#)  
*No such field exception class.*
- class [gldb::TableNoSuchRecord](#)  
*No such record exception class.*
- class [gldb::TableMismatchedRecordLength](#)  
*Mismatched record length 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.*

#### 7.2.1 Detailed Description

Module for interacting with the database.

## 7.2.2 Function Documentation

### 7.2.2.1 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.

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.

### 7.2.2.2 std::string glldb::get\_database\_type ( )

Returns the name of the compiled-in database type.

#### Returns

The name of the compiled-in database type.

## 7.3 SQL statements module

### Classes

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

### 7.3.1 Detailed Description

Module for producing SQL statements used by program.

## 7.4 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.*

#### 7.4.1 Detailed Description

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

## 7.5 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.*

#### 7.5.1 Detailed Description

General purpose helper classes and functions.

#### 7.5.2 Function Documentation

##### 7.5.2.1 `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.

##### 7.5.2.2 `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`.

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

Trims leading and trailing whitespace from a string.

## Parameters

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

## Returns

The trimmed string.

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

Trims trailing whitespace from a string.

## Parameters

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

## Returns

The trimmed string.

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

Trims leading whitespace from a string.

## Parameters

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

## Returns

The trimmed string.

## 7.6 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.*

#### 7.6.1 Detailed Description

Administrative reporting program.

#### 7.6.2 Function Documentation

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

Gets a password from the terminal.

##### Returns

The password.

##### 7.6.2.2 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.

7.6.2.3 `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> .

## 7.7 Database program.

### Functions

- static void `set_configuration` (`Config` &config, int argc, char \*argv[])  
*Sets program configuration options.*
- static bool `check_help_and_version` (const `Config` &config)  
*Prints help or version messages if requested.*
- static bool `check_db_parameters` (const `Config` &config)  
*Checks if database, hostname and username were provided.*
- 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.*

#### 7.7.1 Detailed Description

Administrative database management program.

#### 7.7.2 Function Documentation

**7.7.2.1** static bool `check_db_parameters` ( const `Config` & *config* ) [static]

Checks if database, hostname and username were provided.

##### Parameters

<i>config</i>	Reference to a Config object.
---------------	-------------------------------

##### Returns

`true` if the information was provided, `false` otherwise.

**7.7.2.2** static bool `check_help_and_version` ( const `Config` & *config* ) [static]

Prints help or version messages if requested.

##### Parameters

<i>config</i>	Reference to a Config object.
---------------	-------------------------------

**Returns**

`true` if the help or version message was requested, `false` otherwise.

**7.7.2.3 static std::string login ( void ) [static]**

Gets a password from the terminal.

**Returns**

The password.

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

Main function.

**Parameters**

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

**Returns**

Exit status code.

**7.7.2.5 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 8

# Class Documentation

### 8.1 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](#)

#### 8.1.1 Detailed Description

Configuration options class.

#### 8.1.2 Constructor & Destructor Documentation

##### 8.1.2.1 Config::Config ( )

Constructor

### 8.1.2.2 Config::~Config ( )

Destructor

## 8.1.3 Member Function Documentation

### 8.1.3.1 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.

### 8.1.3.2 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.

### 8.1.3.3 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.
---	---------------------------------------

### 8.1.3.4 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.
--	---

## 8.1.3.5 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.

## 8.1.4 Member Data Documentation

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

Map of options which have been set

## 8.1.4.2 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:

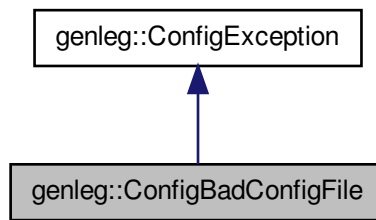
- lib/config/[config.h](#)
- lib/config/[config.cpp](#)
- lib/config/[config\\_getopt.cpp](#)

## 8.2 genleg::ConfigBadConfigFile Class Reference

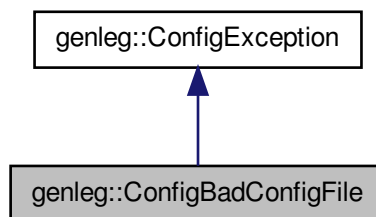
Exception class for badly formed configuration file.

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

Inheritance diagram for `genleg::ConfigBadConfigFile`:



Collaboration diagram for `genleg::ConfigBadConfigFile`:



## Public Member Functions

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

### 8.2.1 Detailed Description

Exception class for badly formed configuration file.

### 8.2.2 Constructor & Destructor Documentation

**8.2.2.1** `genleg::ConfigBadConfigFile::ConfigBadConfigFile ( 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:



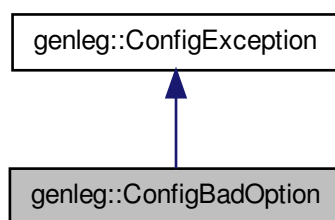
- [lib/config/config.h](#)

## 8.3 genleg::ConfigBadOption Class Reference

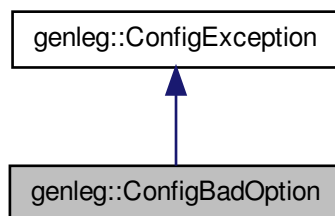
Exception class for bad provided option.

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

Inheritance diagram for genleg::ConfigBadOption:



Collaboration diagram for genleg::ConfigBadOption:



### Public Member Functions

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

#### 8.3.1 Detailed Description

Exception class for bad provided option.

#### 8.3.2 Constructor & Destructor Documentation

### 8.3.2.1 `genleg::ConfigBadOption::ConfigBadOption ( 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:

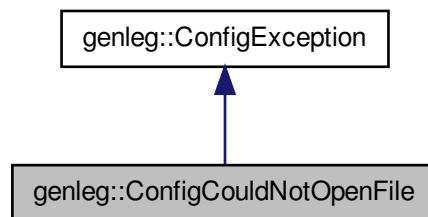
- [lib/config/config.h](#)

## 8.4 `genleg::ConfigCouldNotOpenFile` Class Reference

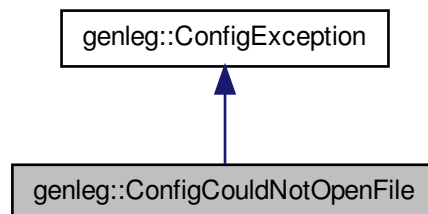
Exception class for when conf file cannot be opened.

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

Inheritance diagram for `genleg::ConfigCouldNotOpenFile`:



Collaboration diagram for `genleg::ConfigCouldNotOpenFile`:



### Public Member Functions

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

### 8.4.1 Detailed Description

Exception class for when conf file cannot be opened.

### 8.4.2 Constructor & Destructor Documentation

8.4.2.1 `genleg::ConfigCouldNotOpenFile::ConfigCouldNotOpenFile ( 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:

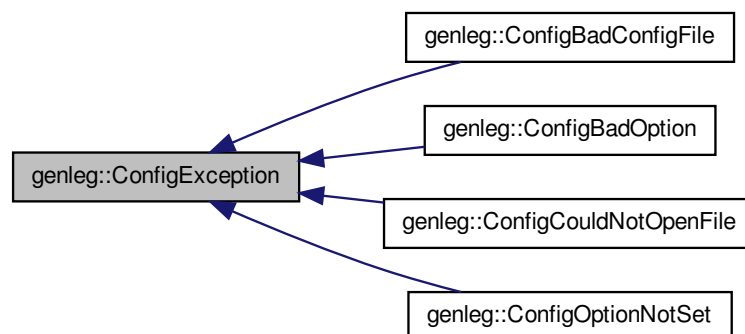
- [lib/config/config.h](#)

## 8.5 genleg::ConfigException Class Reference

Configuration module exception base class.

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

Inheritance diagram for genleg::ConfigException:



### Public Member Functions

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

### 8.5.1 Detailed Description

Configuration module exception base class.

## 8.5.2 Constructor & Destructor Documentation

### 8.5.2.1 `genleg::ConfigException::ConfigException ( 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:

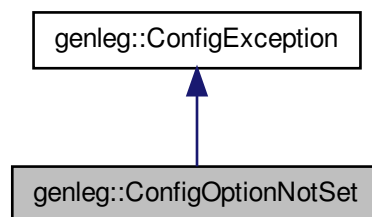
- `lib/config/config.h`

## 8.6 `genleg::ConfigOptionNotSet` Class Reference

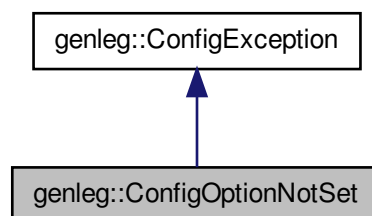
Exception class for option not set.

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

Inheritance diagram for `genleg::ConfigOptionNotSet`:



Collaboration diagram for `genleg::ConfigOptionNotSet`:



## Public Member Functions

- [`ConfigOptionNotSet`](#) (const std::string &msg)

*Constructor.*

### 8.6.1 Detailed Description

Exception class for option not set.

### 8.6.2 Constructor & Destructor Documentation

#### 8.6.2.1 genleg::ConfigOptionNotSet::ConfigOptionNotSet ( 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:

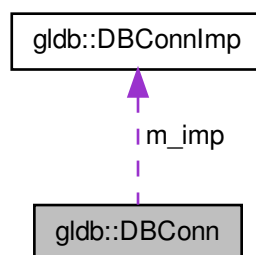
- lib/config/[config.h](#)

## 8.7 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`

### 8.7.1 Detailed Description

Database connection class.

### 8.7.2 Constructor & Destructor Documentation

#### 8.7.2.1 `DBConn::DBConn ( DBConnImp * imp )` `[explicit]`

Constructor.

##### Parameters

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

#### 8.7.2.2 `gldb::DBConn::DBConn ( const DBConn & )`

Deleted copy constructor

### 8.7.3 Member Function Documentation

#### 8.7.3.1 `DBConn& gldb::DBConn::operator= ( const DBConn & )`

Deleted assignment operator

#### 8.7.3.2 `void DBConn::query ( std::string sql_query )`

Runs an SQL query.

##### Parameters

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

##### Returns

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

#### 8.7.3.3 `Table DBConn::select ( std::string query )`

Runs an SQL SELECT query.

##### Parameters

<code>query</code>	The query.
--------------------	------------

### Returns

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

## 8.7.4 Member Data Documentation

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

Pointer to database implementation object.

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

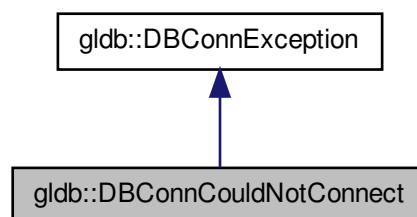
- [lib/database/dbconn.h](#)
- [lib/database/dbconn.cpp](#)

## 8.8 glldb::DBConnCouldNotConnect Class Reference

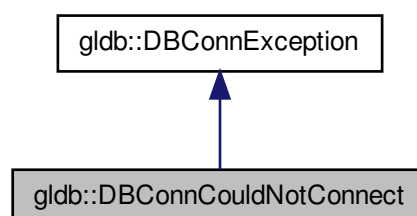
Could not connect to database exception class.

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

Inheritance diagram for glldb::DBConnCouldNotConnect:



Collaboration diagram for glldb::DBConnCouldNotConnect:



## Public Member Functions

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

### 8.8.1 Detailed Description

Could not connect to database exception class.

### 8.8.2 Constructor & Destructor Documentation

**8.8.2.1** `gldb::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:

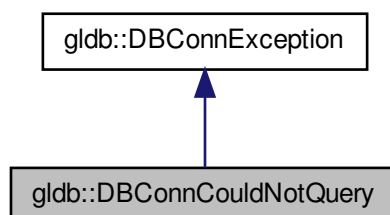
- lib/database/[dbconn.h](#)

## 8.9 gldb::DBConnCouldNotQuery Class Reference

Could not execute database query exception class.

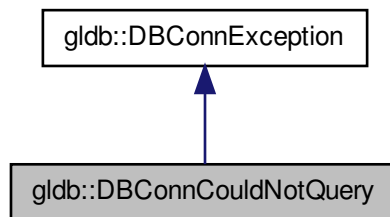
```
#include <dbconn.h>
```

Inheritance diagram for gldb::DBConnCouldNotQuery:





Collaboration diagram for glldb::DBConnCouldNotQuery:



## Public Member Functions

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

### 8.9.1 Detailed Description

Could not execute database query exception class.

### 8.9.2 Constructor & Destructor Documentation

8.9.2.1 `glldb::DBConnCouldNotQuery::DBConnCouldNotQuery ( 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:

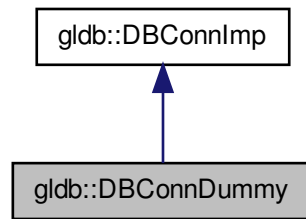
- lib/database/[dbconn.h](#)

## 8.10 glldb::DBConnDummy Class Reference

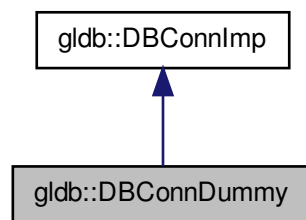
Dummy database implementation class.

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

Inheritance diagram for `gldb::DBConnDummy`:



Collaboration diagram for `gldb::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.*

### 8.10.1 Detailed Description

Dummy database implementation class.

### 8.10.2 Constructor & Destructor Documentation

- 8.10.2.1 `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.

## 8.10.2.2 gldb::DBConnDummy::DBConnDummy ( const DBConnDummy &amp; )

Deleted copy constructor

## 8.10.2.3 DBConnDummy::~~DBConnDummy ( ) [virtual]

Destructor

## 8.10.3 Member Function Documentation

## 8.10.3.1 DBConnDummy&amp; gldb::DBConnDummy::operator= ( const DBConnDummy &amp; )

Deleted assignment operator

## 8.10.3.2 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:

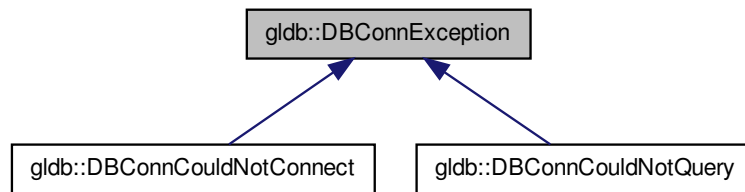
- lib/database\_imp/dummy/dbconn\_dummy\_imp.h
- lib/database\_imp/dummy/dbconn\_dummy\_imp.cpp

## 8.11 gldb::DBConnException Class Reference

Base database connection exception class.

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

Inheritance diagram for `gldb::DBConnException`:



## Public Member Functions

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

### 8.11.1 Detailed Description

Base database connection exception class.

### 8.11.2 Constructor & Destructor Documentation

#### 8.11.2.1 `gldb::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:

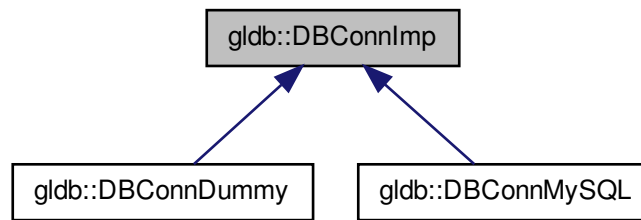
- `lib/database/dbconn.h`

## 8.12 gldb::DBConnImp Class Reference

Abstract database implementation base class.

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

Inheritance diagram for glldb::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.*

### 8.12.1 Detailed Description

Abstract database implementation base class.

### 8.12.2 Constructor & Destructor Documentation

8.12.2.1 `glldb::DBConnImp::DBConnImp ( ) [inline]`

Constructor

8.12.2.2 `virtual glldb::DBConnImp::~~DBConnImp ( ) [inline],[virtual]`

Destructor

### 8.12.3 Member Function Documentation

8.12.3.1 `virtual void glldb::DBConnImp::query ( std::string sql_query ) [pure virtual]`

Runs an SQL query.

Parameters

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

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

### 8.12.3.2 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:

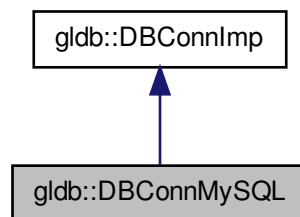
- lib/database/[dbconnimp.h](#)

## 8.13 glldb::DBConnMySQL Class Reference

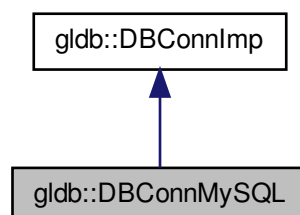
MySQL database implementation class.

```
#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](#)

### 8.13.1 Detailed Description

MySQL database implementation class.

### 8.13.2 Constructor & Destructor Documentation

- 8.13.2.1 [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.
---------------------------------------	-----------------------------------

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

Deleted copy constructor

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

Destructor

### 8.13.3 Member Function Documentation

#### 8.13.3.1 DBConnMySQL& glldb::DBConnMySQL::operator= ( const DBConnMySQL & )

Deleted assignment operator

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

Runs an SQL query.

##### Parameters

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

##### Exceptions

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

Implements [glldb::DBConnImp](#).

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

Implements [glldb::DBConnImp](#).

### 8.13.4 Member Data Documentation

#### 8.13.4.1 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](#)

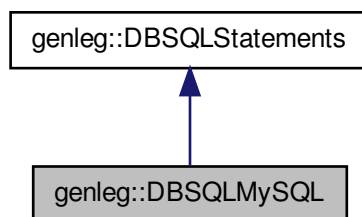
## 8.14 genleg::DBSQLMySQL Class Reference

MySQL SQL statements class.

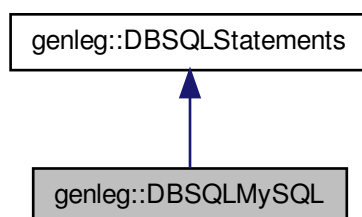
```
#include <dbsql_mysql.h>
```



Inheritance diagram for genleg::DBSQLMySQL:



Collaboration diagram for genleg::DBSQLMySQL:



## Additional Inherited Members

### 8.14.1 Detailed Description

MySQL SQL statements class.

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

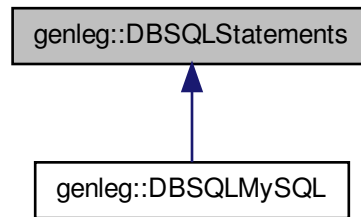
- lib/dbsql/[dbsql\\_mysql.h](#)

## 8.15 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.
- virtual std::string [user\\_by\\_id](#) (const std::string &user\_id) const  
Returns a SQL statement to select a user by ID.
- virtual std::string [user\\_by\\_username](#) (const std::string &user\_name) const  
Returns a SQL statement to select a user by username.
- virtual std::string [update\\_user](#) (const [GLUser](#) &user) const  
Returns a SQL UPDATE statement to update a user.
- virtual std::string [grant](#) (const std::string &user\_id, const std::string &perm) const  
Returns a SQL statement to grant a user a permission.
- virtual std::string [revoke](#) (const std::string &user\_id, const std::string &perm) const  
Returns a SQL UPDATE statement to revoke a permission from a user.
- virtual std::string [get\\_perms](#) (const std::string &user\_id) const  
Returns a SQL UPDATE statement to list a user's permissions.

### 8.15.1 Detailed Description

SQL statements class.

### 8.15.2 Constructor & Destructor Documentation

#### 8.15.2.1 DBSQLStatements::DBSQLStatements ( )

Constructor

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

Destructor

## 8.15.3 Member Function Documentation

### 8.15.3.1 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.

### 8.15.3.2 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.

### 8.15.3.3 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.

### 8.15.3.4 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.

**8.15.3.5** `std::string DBSQLStatements::get_perms ( const std::string & user_id ) const` [virtual]

Returns a SQL UPDATE statement to list a user's permissions.

#### Parameters

<i>user_id</i>	The user ID for which to list.
----------------	--------------------------------

#### Returns

The SQL statement.

**8.15.3.6** `std::string DBSQLStatements::grant ( const std::string & user_id, const std::string & perm ) const` [virtual]

Returns a SQL statement to grant a user a permission.

#### Attention

This function always sets the user granting the permission to user 1. This will need to be updated to support the recording of which user has granted the permission, when support for others to be able to do so is implemented.

#### Parameters

<i>user_id</i>	The user ID for which to grant the permission.
<i>perm</i>	A string containing the name of the permission.

#### Returns

The SQL statement.

**8.15.3.7** `std::string DBSQLStatements::revoke ( const std::string & user_id, const std::string & perm ) const` [virtual]

Returns a SQL UPDATE statement to revoke a permission from a user.

#### Parameters

<i>user_id</i>	The user ID from which to revoke.
<i>perm</i>	The permission to revoke.

#### Returns

The SQL statement.

**8.15.3.8** `std::string DBSQLStatements::update_user ( const GLUser & user ) const` [virtual]

Returns a SQL UPDATE statement to update a user.

#### Parameters

<i>user</i>	A user object.
-------------	----------------

**Returns**

The SQL statement.

8.15.3.9 `std::string DBSQLStatements::user_by_id ( const std::string & user_id ) const` [virtual]

Returns a SQL statement to select a user by ID.

**Parameters**

<i>user_id</i>	The user_id
----------------	-------------

**Returns**

The SQL statement.

8.15.3.10 `std::string DBSQLStatements::user_by_username ( const std::string & user_name ) const` [virtual]

Returns a SQL statement to select a user by username.

**Parameters**

<i>user_name</i>	The username.
------------------	---------------

**Returns**

The SQL statement.

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

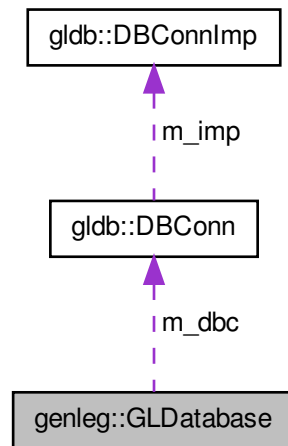
- [lib/dbsql/dbsqlstatements.h](#)
- [lib/dbsql/dbsqlstatements.cpp](#)

## 8.16 genleg::GLDatabase Class Reference

General ledger database class.

```
#include <gldatabase.h>
```

Collaboration diagram for genleg::GLDatabase:



## Public Member Functions

- `GLDatabase` (const std::string &database, const std::string &hostname, const std::string &username, const std::string &password)  
*Constructor.*
- `~GLDatabase` ()
- void `create_structure` ()  
*Creates the database structure.*
- void `destroy_structure` ()  
*Destroys the database structure.*
- void `load_sample_data` (const std::string &dir)  
*Loads sample data into the database.*
- `GLUser` `get_user_by_id` (const std::string &user\_id)  
*Returns a user from an ID.*
- `GLUser` `get_user_by_username` (const std::string &user\_name)  
*Returns a user from a user name.*
- void `update_user` (const `GLUser` &user)  
*Updates a user's details.*
- void `grant` (const `GLUser` &user, const std::string &perm)  
*Grants a user a permission.*
- void `revoke` (const `GLUser` &user, const std::string &perm)  
*Revokes a permission from a user.*

## Static Public Member Functions

- static std::string `backend` ()  
*Returns the backend database implementation.*

## Private Member Functions

- [GLUser create\\_user](#) ([gldb::Table](#) &table)  
*Creates a user from a query table.*

## Private Attributes

- [gldb::DBConn m\\_dbc](#)
- `const std::shared_ptr< const DBSQLStatements > m_sql`
- `const std::vector< std::string > m_tables`
- `const std::vector< std::string > m_views`

### 8.16.1 Detailed Description

General ledger database class.

### 8.16.2 Constructor & Destructor Documentation

8.16.2.1 `GLDatabase::GLDatabase ( const std::string & database, const std::string & hostname, const std::string & username, const std::string & password )`

Constructor.

#### Parameters

<i>database</i>	Database name.
<i>hostname</i>	Hostname of database machine.
<i>username</i>	Username to log into database.
<i>password</i>	Password to log into database.

#### Exceptions

<a href="#">GLDBException</a>	on error.
-------------------------------	-----------

8.16.2.2 `GLDatabase::~~GLDatabase ( )`

Destructor

### 8.16.3 Member Function Documentation

8.16.3.1 `std::string GLDatabase::backend ( ) [static]`

Returns the backend database implementation.

This may be called to discover which database platform support has been compiled into the application.

#### Returns

A string containing the database platform name.

### 8.16.3.2 void GLDatabase::create\_structure ( )

Creates the database structure.

#### Exceptions

<a href="#"><i>GLDBException</i></a>	on error.
--------------------------------------	-----------

### 8.16.3.3 GLUser GLDatabase::create\_user ( glldb::Table & table ) [private]

Creates a user from a query table.

Provided because the public functions can get a user either from an ID or a name, this function contains the common functionality.

#### Parameters

<i>table</i>	A table from the appropriate query.
--------------	-------------------------------------

#### Returns

The new user.

### 8.16.3.4 void GLDatabase::destroy\_structure ( )

Destroys the database structure.

#### Exceptions

<a href="#"><i>GLDBException</i></a>	on error.
--------------------------------------	-----------

### 8.16.3.5 GLUser GLDatabase::get\_user\_by\_id ( const std::string & user\_id )

Returns a user from an ID.

#### Parameters

<i>user_id</i>	The user ID.
----------------	--------------

#### Returns

The user.

#### Exceptions

<a href="#"><i>GLDBException</i></a>	if the user cannot be found.
--------------------------------------	------------------------------

### 8.16.3.6 GLUser GLDatabase::get\_user\_by\_username ( const std::string & user\_name )

Returns a user from a user name.



## Parameters

<i>user_name</i>	The user name.
------------------	----------------

## Returns

The user.

## Exceptions

<a href="#"><i>GLDBException</i></a>	if the user cannot be found.
--------------------------------------	------------------------------

8.16.3.7 void GLDatabase::grant ( const GLUser & *user*, const std::string & *perm* )

Grants a user a permission.

## Parameters

<i>user</i>	The user for which to grant.
<i>perm</i>	A string containing the permission to grant.

8.16.3.8 void GLDatabase::load\_sample\_data ( const std::string & *dir* )

Loads sample data into the database.

## Parameters

<i>dir</i>	The directory containing the sample data. Individual files in that directory should be named after the table they are intended to populate.
------------	---

## Exceptions

<a href="#"><i>GLDBException</i></a>	on error.
--------------------------------------	-----------

8.16.3.9 void GLDatabase::revoke ( const GLUser & *user*, const std::string & *perm* )

Revokes a permission from a user.

## Parameters

<i>user</i>	The user for which to revoke.
<i>perm</i>	A string containing the permission to revoke.

8.16.3.10 void GLDatabase::update\_user ( const GLUser & *user* )

Updates a user's details.

## Parameters

<i>user</i>	The user object.
-------------	------------------

## 8.16.4 Member Data Documentation

#### 8.16.4.1 `gldb::DBConn` `genleg::GLDatabase::m_dbc` [private]

Database connection

#### 8.16.4.2 `const std::shared_ptr<const DBSQLStatements>` `genleg::GLDatabase::m_sql` [private]

SQL statements object

#### 8.16.4.3 `const std::vector<std::string>` `genleg::GLDatabase::m_tables` [private]

Vector containing database table names

#### 8.16.4.4 `const std::vector<std::string>` `genleg::GLDatabase::m_views` [private]

Vector containing database view names

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

- [lib/gldb/gldatabase.h](#)
- [lib/gldb/gldatabase.cpp](#)

## 8.17 `genleg::GLDBException` Class Reference

Base general ledger database exceptionc class.

```
#include <glexception.h>
```

### Public Member Functions

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

#### 8.17.1 Detailed Description

Base general ledger database exceptionc class.

#### 8.17.2 Constructor & Destructor Documentation

##### 8.17.2.1 `genleg::GLDBException::GLDBException ( 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:

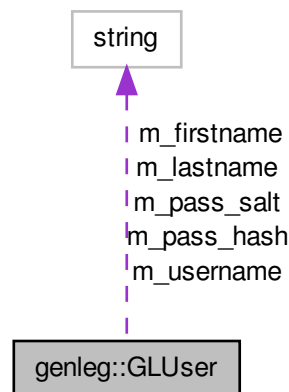
- [lib/gldb/glexception.h](#)

## 8.18 genleg::GLUser Class Reference

General ledger user class.

```
#include <gluser.h>
```

Collaboration diagram for genleg::GLUser:



### Public Member Functions

- [GLUser](#) (const std::string &[id](#), const std::string &[username](#), const std::string &[firstname](#), const std::string &[lastname](#), const std::string &[pass\\_hash](#), const std::string &[pass\\_salt](#), std::vector< std::string > &&[perms](#), const bool [enabled](#))  
*Constructor.*
- [~GLUser](#) ()
- const std::string & [id](#) () const  
*Returns the user ID.*
- const std::string & [username](#) () const  
*Returns the username.*
- const std::string & [firstname](#) () const  
*Returns the user's first name.*
- const std::string & [lastname](#) () const  
*Returns the user's last name.*
- const std::string & [pass\\_hash](#) () const  
*Returns the user's hashed password.*
- const std::string & [pass\\_salt](#) () const  
*Returns the user's password salt.*
- const std::vector< std::string > & [permissions](#) () const  
*Returns the permissions for a user.*
- bool [enabled](#) () const  
*Returns the user's enabled status.*
- void [set\\_username](#) (const std::string &[new\\_username](#))  
*Sets a user's username.*
- void [set\\_firstname](#) (const std::string &[new\\_firstname](#))

*Sets a user's first name.*

- void `set_lastname` (const std::string &new\_lastname)

*Sets a user's last name.*

- void `set_enabled` (const bool new\_enabled)

*Sets a user's enabled status.*

- void `set_password` (const std::string &new\_pass)

*Sets a user's password hash and salt.*

- bool `check_password` (const std::string &check\_pass)

*Checks a password against the user's hash.*

## Private Attributes

- const std::string `m_id`
- std::string `m_username`
- std::string `m_firstname`
- std::string `m_lastname`
- std::string `m_pass_hash`
- std::string `m_pass_salt`
- const std::vector< std::string > `m_perms`
- bool `m_enabled`

### 8.18.1 Detailed Description

General ledger user class.

### 8.18.2 Constructor & Destructor Documentation

- 8.18.2.1 `GLUser::GLUser ( const std::string & id, const std::string & username, const std::string & firstname, const std::string & lastname, const std::string & pass_hash, const std::string & pass_salt, std::vector< std::string > && perms, const bool enabled )`

Constructor.

#### Parameters

<i>id</i>	User ID
<i>username</i>	Username
<i>firstname</i>	First name
<i>lastname</i>	Last name
<i>pass_hash</i>	The hashed password
<i>pass_salt</i>	The salt for the hashed password
<i>perms</i>	Vector of user permissions
<i>enabled</i>	true if user is enabled, false otherwise.

- 8.18.2.2 `GLUser::~~GLUser ( )`

Destructor

### 8.18.3 Member Function Documentation

**8.18.3.1** `bool GLUser::check_password ( const std::string & check_pass )`

Checks a password against the user's hash.

**Parameters**

<i>check_pass</i>	The password to check, must be > 8 characters.
-------------------	--

**Returns**

`true` is the password matches, `false` otherwise.

**8.18.3.2** `bool GLUser::enabled ( ) const`

Returns the user's enabled status.

**Returns**

The user's enabled status.

**8.18.3.3** `const std::string & GLUser::firstname ( ) const`

Returns the user's first name.

**Returns**

The user's first name.

**8.18.3.4** `const std::string & GLUser::id ( ) const`

Returns the user ID.

**Returns**

The user ID.

**8.18.3.5** `const std::string & GLUser::lastname ( ) const`

Returns the user's last name.

**Returns**

The user's last name.

**8.18.3.6** `const std::string & GLUser::pass_hash ( ) const`

Returns the user's hashed password.

**Returns**

The user's hashed password.

8.18.3.7 `const std::string & GLUser::pass_salt ( ) const`

Returns the user's password salt.

Returns

The user's password salt.

8.18.3.8 `const std::vector< std::string > & GLUser::permissions ( ) const`

Returns the permissions for a user.

Returns

A vector of strings containing the names of the permissions held by the user.

8.18.3.9 `void GLUser::set_enabled ( const bool new_enabled )`

Sets a user's enabled status.

Parameters

<i>new_enabled</i>	The user's new enabled status.
--------------------	--------------------------------

8.18.3.10 `void GLUser::set_firstname ( const std::string & new_firstname )`

Sets a user's first name.

Parameters

<i>new_firstname</i>	The user's new first name.
----------------------	----------------------------

8.18.3.11 `void GLUser::set_lastname ( const std::string & new_lastname )`

Sets a user's last name.

Parameters

<i>new_lastname</i>	The user's new last name.
---------------------	---------------------------

8.18.3.12 `void GLUser::set_password ( const std::string & new_pass )`

Sets a user's password hash and salt.

Parameters

<i>new_pass</i>	The new password, must be > 8 characters.
-----------------	---

8.18.3.13 `void GLUser::set_username ( const std::string & new_username )`

Sets a user's username.

## Parameters

<i>new_username</i>	The user's new username.
---------------------	--------------------------

8.18.3.14 `const std::string & GLUser::username ( ) const`

Returns the username.

## Returns

The username.

## 8.18.4 Member Data Documentation

8.18.4.1 `bool genleg::GLUser::m_enabled [private]`

User's enabled status

8.18.4.2 `std::string genleg::GLUser::m_firstname [private]`

User's first name

8.18.4.3 `const std::string genleg::GLUser::m_id [private]`

User ID

8.18.4.4 `std::string genleg::GLUser::m_lastname [private]`

User's last name

8.18.4.5 `std::string genleg::GLUser::m_pass_hash [private]`

User's hashed password

8.18.4.6 `std::string genleg::GLUser::m_pass_salt [private]`

User's password salt

8.18.4.7 `const std::vector<std::string> genleg::GLUser::m_perms [private]`

List of permissions

8.18.4.8 `std::string genleg::GLUser::m_username [private]`

Username

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

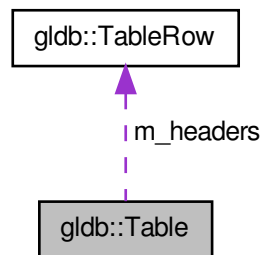
- [lib/gldb/gluser.h](#)
- [lib/gldb/gluser.cpp](#)
- [lib/gldb/gluser\\_pass.cpp](#)

## 8.19 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.*
- std::string [get\\_field](#) (const std::string field\_name, const [size\\_t](#) row\_index)  
*Gets a field from a record by field name.*

### 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`

## 8.19.1 Detailed Description

Database table class.

## 8.19.2 Constructor & Destructor Documentation

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

Constructor.

#### Parameters

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

### 8.19.2.2 `Table::~~Table ( )`

Destructor

## 8.19.3 Member Function Documentation

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

Appends a record to the table.

#### Parameters

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

### 8.19.3.2 `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.

### 8.19.3.3 `std::string Table::get_field ( const std::string field_name, const size_t row_index )`

Gets a field from a record by field name.

#### Parameters

<i>field_name</i>	The name of the field.
<i>row_index</i>	The index of the row.

#### Returns

The contents of the field.

#### Exceptions

<a href="#"><i>TableNoSuchField</i></a>	if <i>field_name</i> is not a valid field name.
<a href="#"><i>TableNoSuchRecord</i></a>	if there is no record at index <i>row_index</i> .

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

Returns the field names.

#### Returns

The field names.

### 8.19.3.5 `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.

### 8.19.3.6 `size_t Table::num_fields ( ) const`

Returns the number of fields in each row.

#### Returns

The number of fields in each row.

### 8.19.3.7 `size_t Table::num_records ( ) const`

Returns the number of record in the table.

#### Returns

The number of records in the table.

8.19.3.8 `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.

8.19.3.9 `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.
------------	---

## 8.19.4 Member Data Documentation

8.19.4.1 `TableRow glldb::Table::m_headers` `[private]`

The names of the fields

8.19.4.2 `std::vector<bool> glldb::Table::m_quoted` `[private]`

A vector to show if fields should be quoted for INSERT

8.19.4.3 `std::vector<TableRow> glldb::Table::m_records` `[private]`

A vector of the records

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

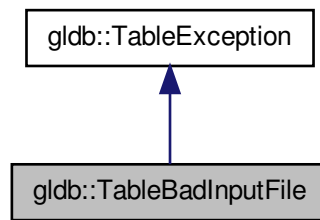
- lib/database/[table.h](#)
- lib/database/[table.cpp](#)

## 8.20 glldb::TableBadInputFile Class Reference

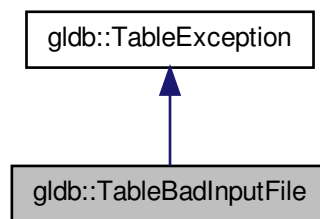
Could not connect to database exception class.

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

Inheritance diagram for `gldb::TableBadInputFile`:



Collaboration diagram for `gldb::TableBadInputFile`:



## Public Member Functions

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

### 8.20.1 Detailed Description

Could not connect to database exception class.

### 8.20.2 Constructor & Destructor Documentation

8.20.2.1 `gldb::TableBadInputFile::TableBadInputFile ( 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:

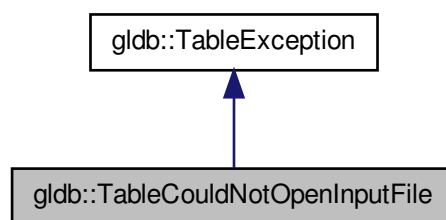
- [lib/database/table.h](#)

## 8.21 glldb::TableCouldNotOpenInputFile Class Reference

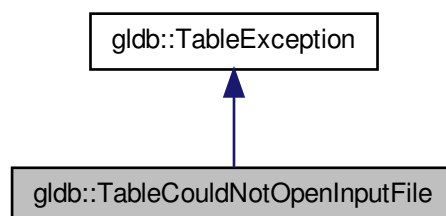
Could not connect to database exception class.

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

Inheritance diagram for glldb::TableCouldNotOpenInputFile:



Collaboration diagram for glldb::TableCouldNotOpenInputFile:



### Public Member Functions

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

#### 8.21.1 Detailed Description

Could not connect to database exception class.

#### 8.21.2 Constructor & Destructor Documentation

8.21.2.1 `gldb::TableCouldNotOpenInputFile::TableCouldNotOpenInputFile ( 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:

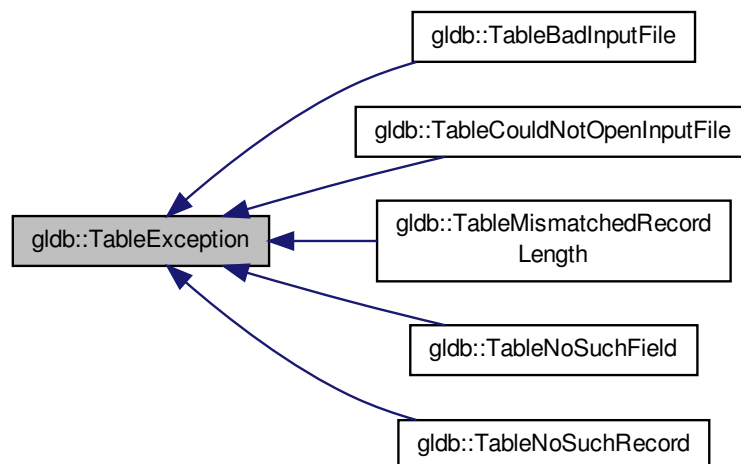
- [lib/database/table.h](#)

## 8.22 gldb::TableException Class Reference

Base database connection exception class.

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

Inheritance diagram for `gldb::TableException`:



### Public Member Functions

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

### 8.22.1 Detailed Description

Base database connection exception class.

### 8.22.2 Constructor & Destructor Documentation

### 8.22.2.1 glldb::TableException::TableException ( 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:

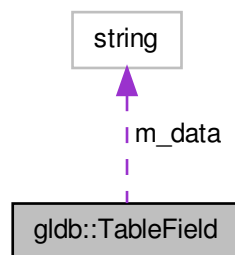
- lib/database/[table.h](#)

## 8.23 glldb::TableField Class Reference

Database table field class.

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

Collaboration diagram for glldb::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.*

## 8.23.1 Detailed Description

Database table field class.

## 8.23.2 Constructor & Destructor Documentation

### 8.23.2.1 [TableField::TableField](#) ( const char \* *data* ) [explicit]

Constructor accepting `const char * data`.

#### Parameters

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

### 8.23.2.2 [TableField::TableField](#) ( const std::string & *data* ) [explicit]

Constructor accepting `std::string data`.

#### Parameters

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

### 8.23.2.3 [TableField::~~TableField](#) ( )

Destructor

## 8.23.3 Member Function Documentation

### 8.23.3.1 `size_t` [TableField::length](#) ( ) const

Returns the length of the field.

#### Returns

The length of the field.



### 8.23.3.2 TableField::operator std::string ( ) const

Overridden conversion operator.

Returns the field contents as a string.

### 8.23.3.3 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.

### 8.23.3.4 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.

### 8.23.3.5 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.

### 8.23.3.6 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.

#### 8.23.3.7 `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.

#### 8.23.3.8 `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.

### 8.23.4 Friends And Related Function Documentation

#### 8.23.4.1 `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`.

### 8.23.5 Member Data Documentation

#### 8.23.5.1 `std::string glldb::TableField::m_data` `[private]`

The field contents

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

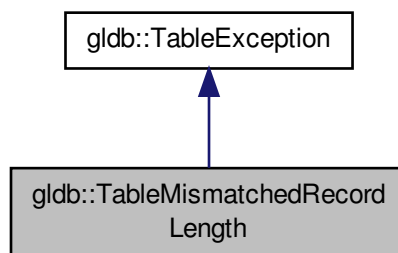
- lib/database/[tablefield.h](#)
- lib/database/[tablefield.cpp](#)

## 8.24 `glldb::TableMismatchedRecordLength` Class Reference

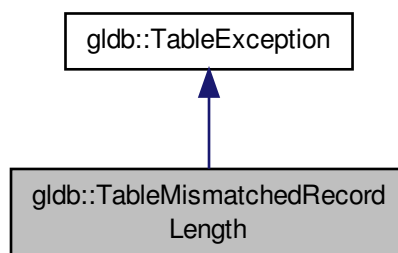
Mismatched record length exception class.

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

Inheritance diagram for glldb::TableMismatchedRecordLength:



Collaboration diagram for glldb::TableMismatchedRecordLength:



## Public Member Functions

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

### 8.24.1 Detailed Description

Mismatched record length exception class.

### 8.24.2 Constructor & Destructor Documentation

**8.24.2.1** `glldb::TableMismatchedRecordLength::TableMismatchedRecordLength ( 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:

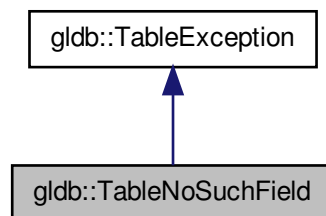
- lib/database/[table.h](#)

## 8.25 glldb::TableNoSuchField Class Reference

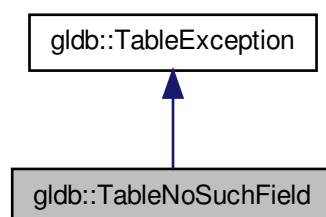
No such field exception class.

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

Inheritance diagram for glldb::TableNoSuchField:



Collaboration diagram for glldb::TableNoSuchField:



### Public Member Functions

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

#### 8.25.1 Detailed Description

No such field exception class.

## 8.25.2 Constructor & Destructor Documentation

### 8.25.2.1 glldb::TableNoSuchField::TableNoSuchField ( 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:

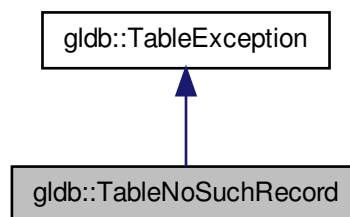
- lib/database/[table.h](#)

## 8.26 glldb::TableNoSuchRecord Class Reference

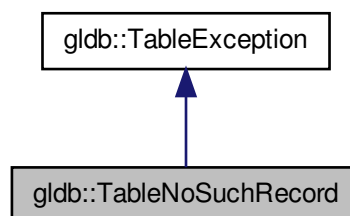
No such record exception class.

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

Inheritance diagram for glldb::TableNoSuchRecord:



Collaboration diagram for glldb::TableNoSuchRecord:



## Public Member Functions

- [TableNoSuchRecord](#) (const std::string &msg)

*Constructor.*

### 8.26.1 Detailed Description

No such record exception class.

### 8.26.2 Constructor & Destructor Documentation

#### 8.26.2.1 `gldb::TableNoSuchRecord::TableNoSuchRecord ( 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/[table.h](#)

## 8.27 `gldb::TableRow` Class Reference

Database table row class.

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

### Public Member Functions

- [TableRow](#) ()
- [TableRow](#) (const size\_t [size](#))  
*Constructor with initial number of fields.*
- [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`

### 8.27.1 Detailed Description

Database table row class.

### 8.27.2 Constructor & Destructor Documentation

#### 8.27.2.1 `TableRow::TableRow ( )`

Default constructor

#### 8.27.2.2 `TableRow::TableRow ( const size_t size )` `[explicit]`

Constructor with initial number of fields.

##### Parameters

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

#### 8.27.2.3 `TableRow::TableRow ( std::vector< std::string > & vec )` `[explicit]`

Constructor with string vector.

##### Parameters

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

#### 8.27.2.4 `TableRow::~~TableRow ( )`

Destructor

### 8.27.3 Member Function Documentation

#### 8.27.3.1 `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.
------------------	--------------------------------

#### 8.27.3.2 `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.
------------------	--------------------------------

**8.27.3.3 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.
------------------	-----------------------------

**8.27.3.4 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.

**8.27.3.5 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.

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

Prints a row.

**Parameters**

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

**8.27.3.7 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.



#### 8.27.3.8 std::string TableRow::record\_string ( )

Creates an unquoted comma separated string of fields.

##### Returns

The unquoted comma separated string.

#### 8.27.3.9 size\_t TableRow::size ( ) const

Returns the number of fields.

##### Returns

The number of fields.

### 8.27.4 Member Data Documentation

#### 8.27.4.1 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](#)



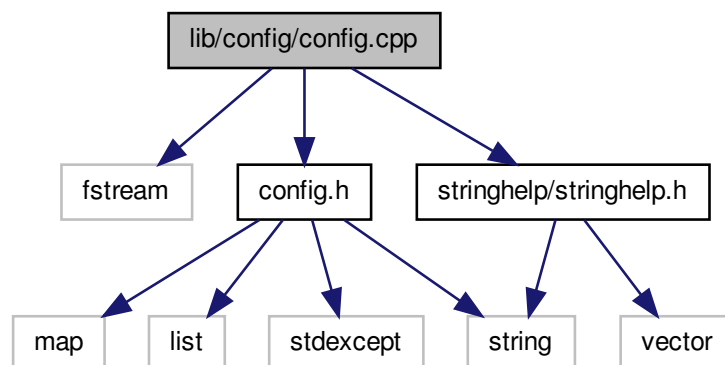
## Chapter 9

# File Documentation

### 9.1 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:
```



#### 9.1.1 Detailed Description

Implementation of program configurations class.

Author

Paul Griffiths

Copyright

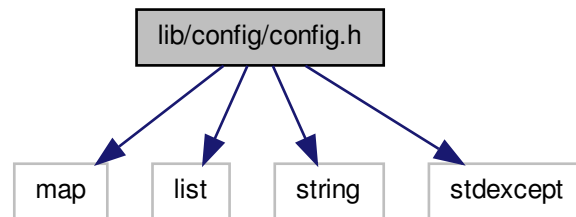
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

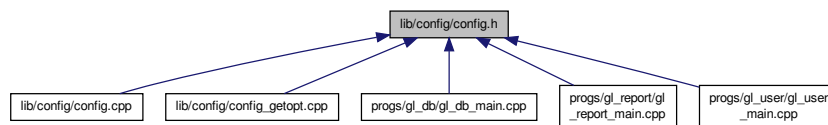
Interface to program configurations class.

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

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.*

### 9.2.1 Detailed Description

Interface to program configurations class.

**Author**

Paul Griffiths

**Copyright**

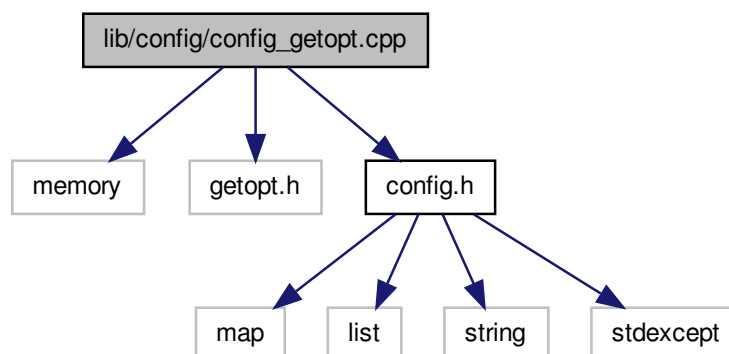
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 9.3 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`

### 9.3.1 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/>

## 9.3.2 Macro Definition Documentation

### 9.3.2.1 `#define _XOPEN_SOURCE 600`

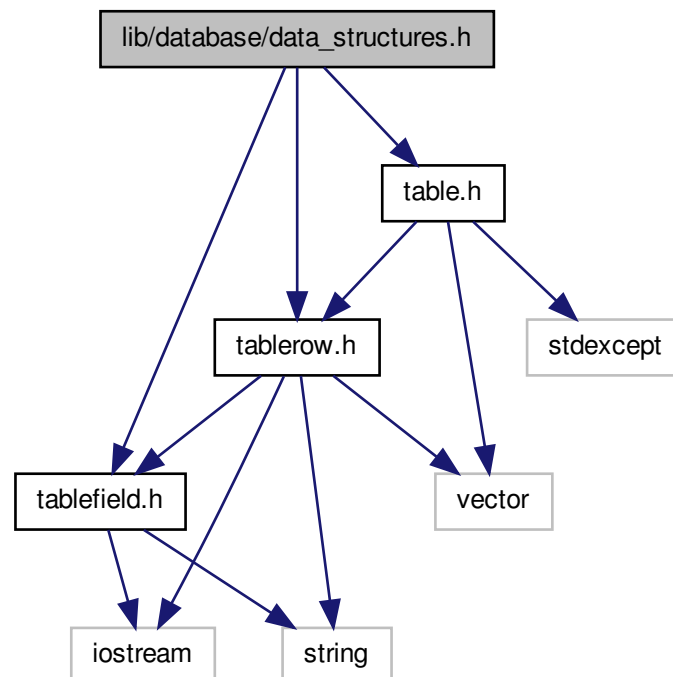
UNIX feature test macro for getopt library

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

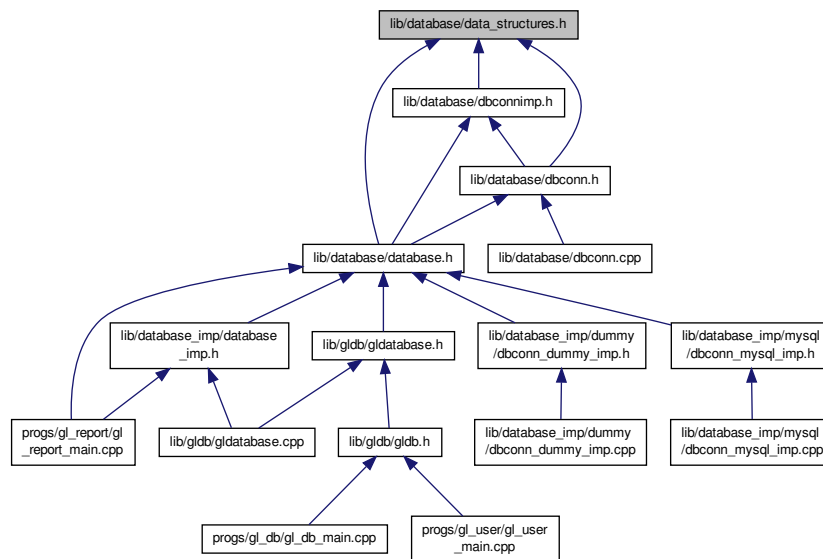
Main interface to database data structures.

```
#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:



### 9.4.1 Detailed Description

Main interface to database data structures.

#### Author

Paul Griffiths

#### Copyright

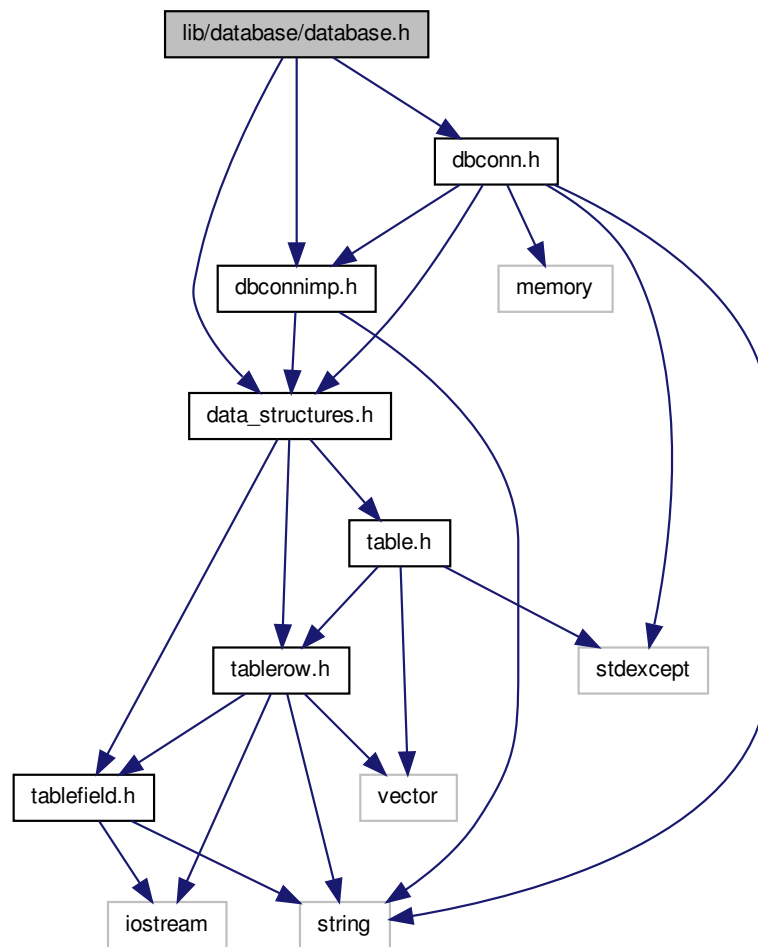
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

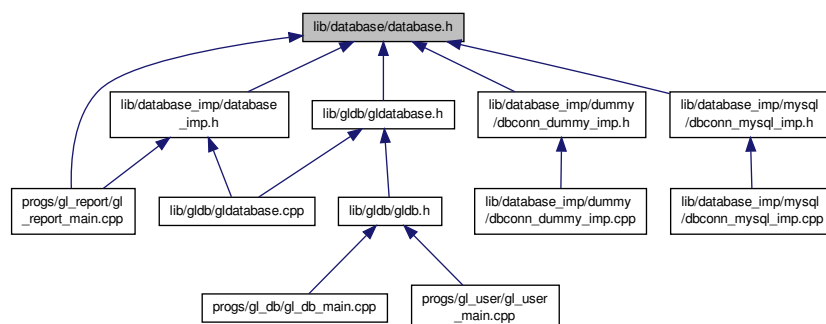
User interface to database functionality.

```
#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:





### 9.5.1 Detailed Description

User interface to database functionality.

Author

Paul Griffiths

Copyright

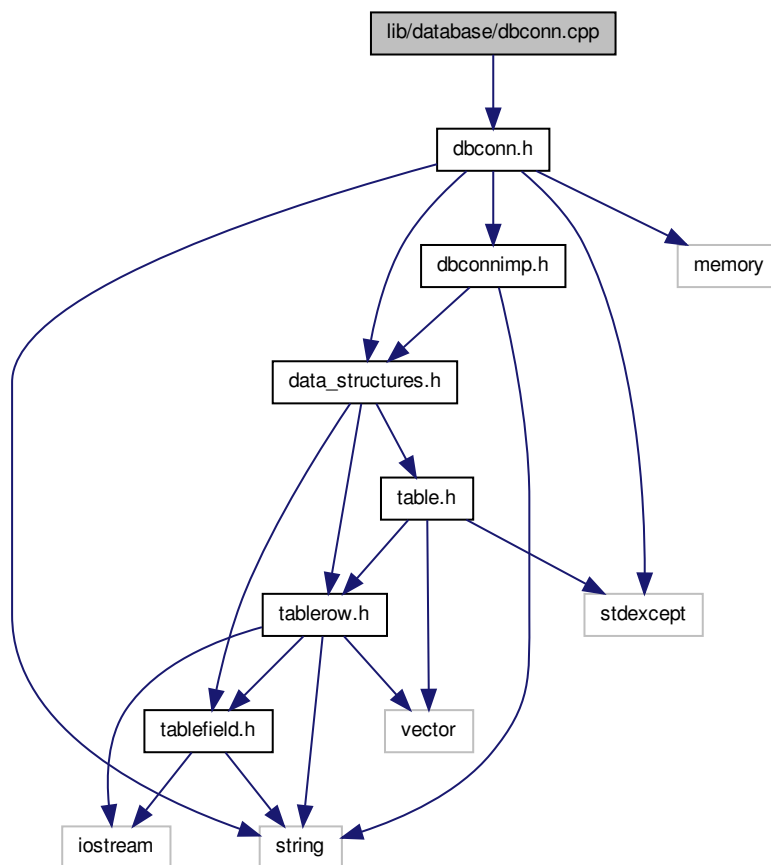
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

Implementation of database connection class.

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

Include dependency graph for dbconn.cpp:



### 9.6.1 Detailed Description

Implementation of database connection class.

## Author

Paul Griffiths

## Copyright

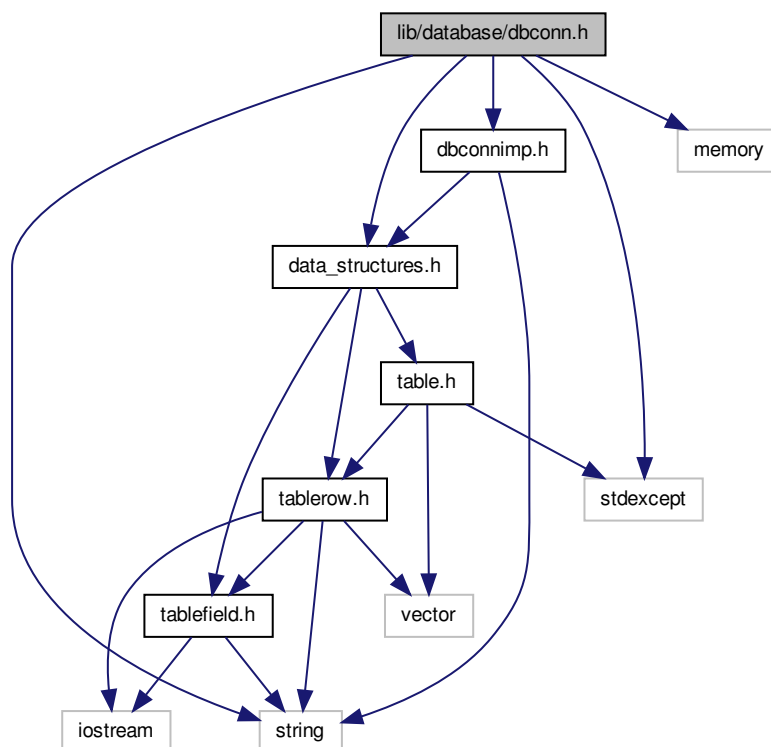
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

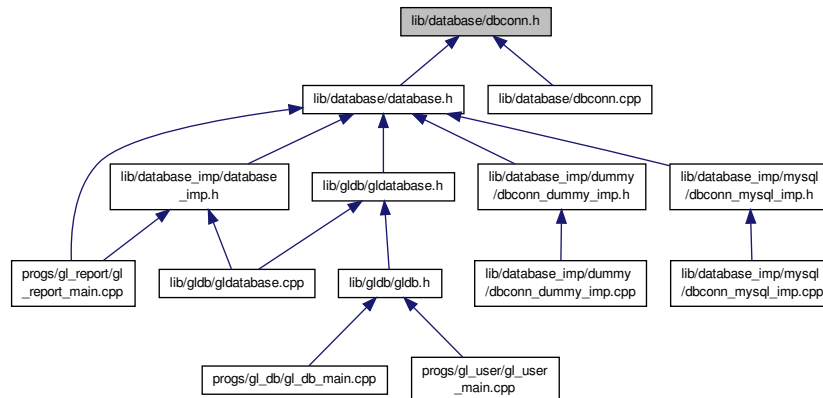
Interface to database connection base class.

```
#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.*

### 9.7.1 Detailed Description

Interface to database connection base class.

#### Author

Paul Griffiths

#### Copyright

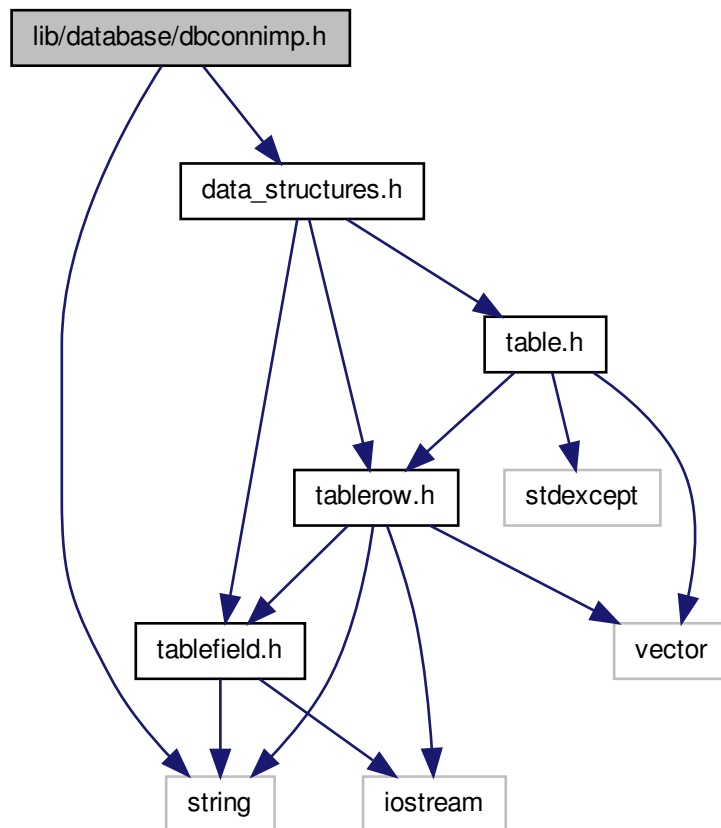
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

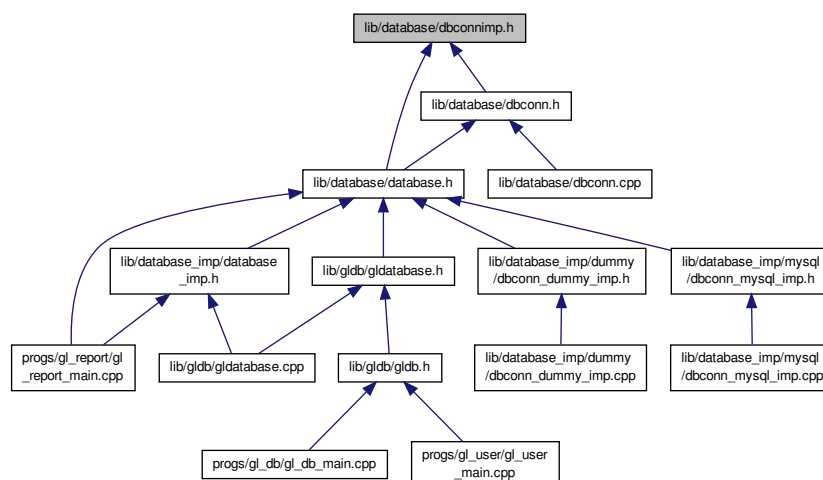
Interface to abstract database implementation base class.

```
#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.*

### 9.8.1 Detailed Description

Interface to abstract database implementation base class.

#### Author

Paul Griffiths

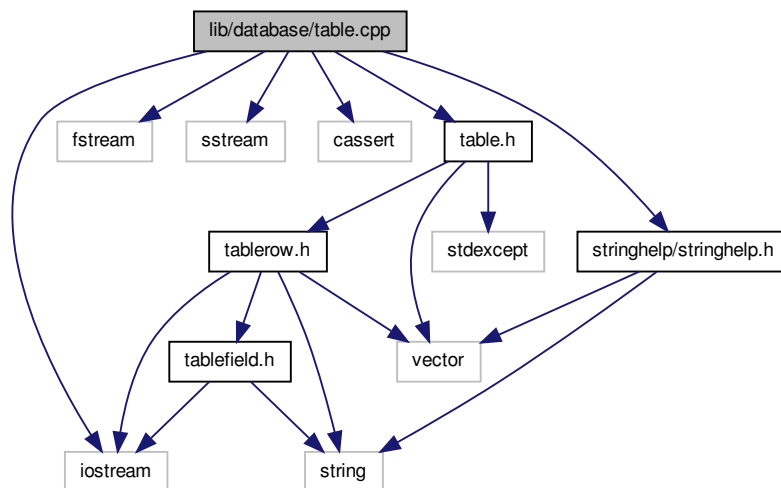
#### Copyright

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

## 9.9 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:
```



### 9.9.1 Detailed Description

Implementation of database table data structure.

## Author

Paul Griffiths

## Copyright

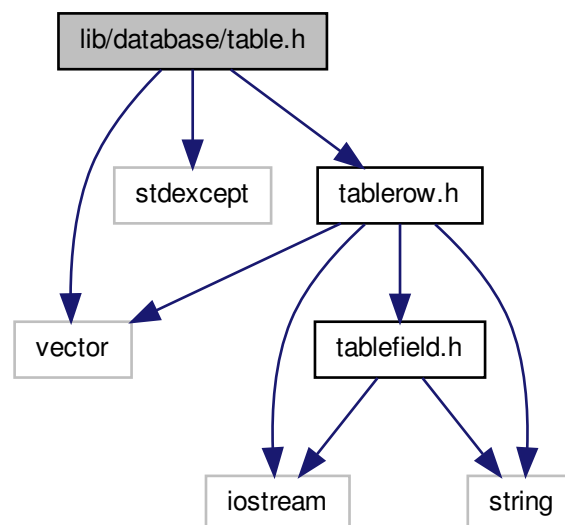
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

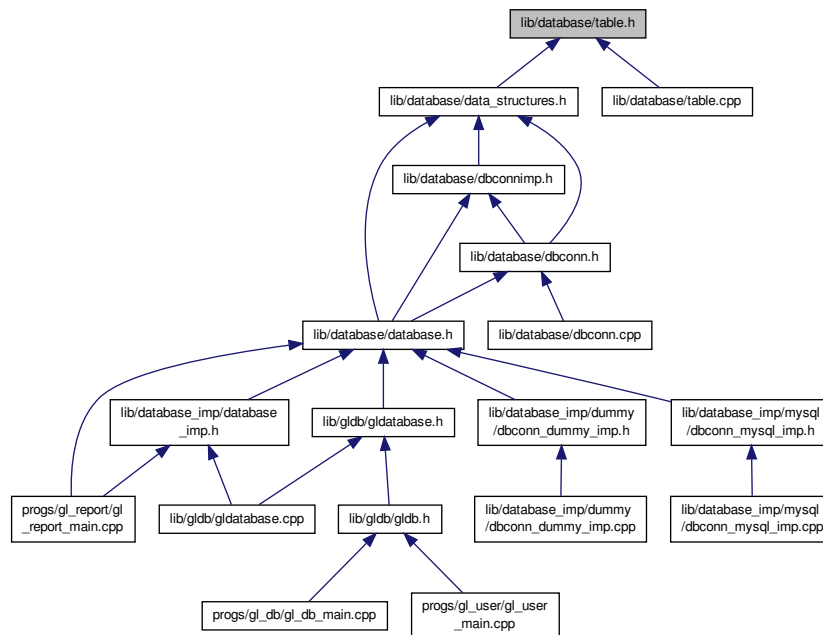
Interface to database table data structure.

```
#include <vector>
#include <stdexcept>
#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::TableNoSuchField](#)  
*No such field exception class.*
- class [gldb::TableNoSuchRecord](#)  
*No such record exception class.*
- class [gldb::TableMismatchedRecordLength](#)  
*Mismatched record length 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.*

### 9.10.1 Detailed Description

Interface to database table data structure.

Author

Paul Griffiths

Copyright

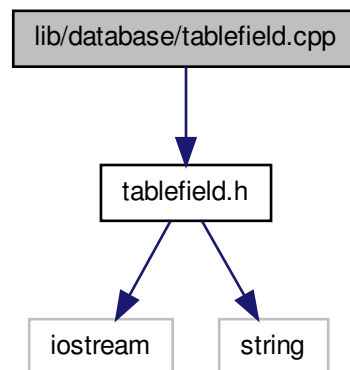
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

Implementation of database table field class.

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

Include dependency graph for tablefield.cpp:



### 9.11.1 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/>

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

Interface to database table field class.

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





Overridden << operator for printing a field.

### 9.12.1 Detailed Description

Interface to database table field class.

#### Author

Paul Griffiths

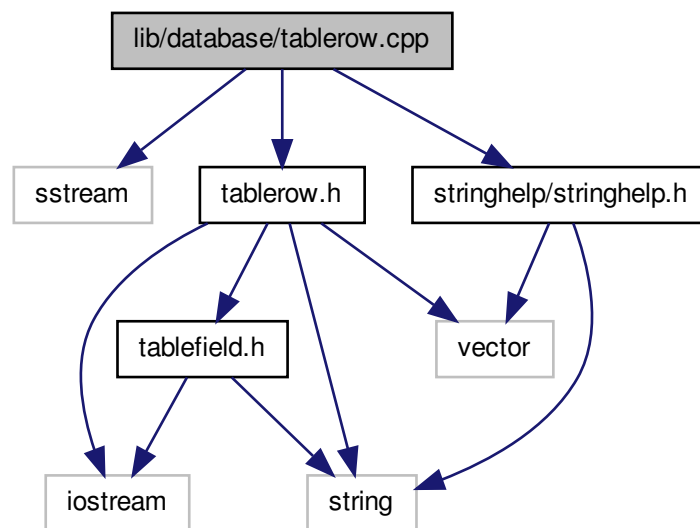
#### Copyright

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

## 9.13 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:
```



### 9.13.1 Detailed Description

Implementation of database table row data structure.

#### Author

Paul Griffiths

### Copyright

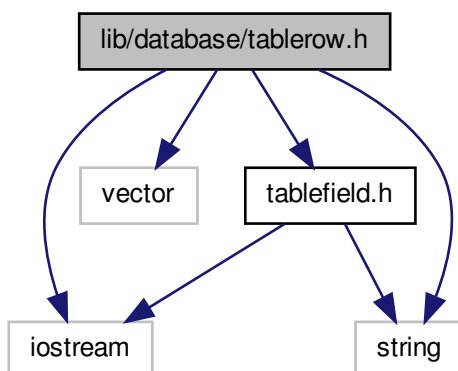
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

Interface to database table row data structure.

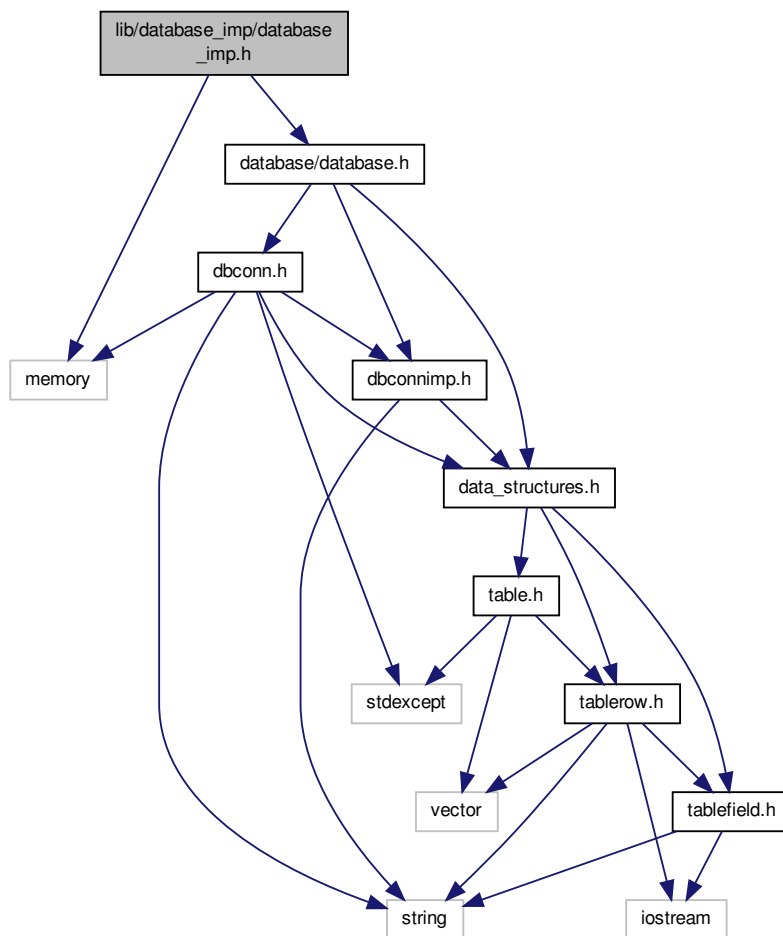
```
#include <iostream>
#include <vector>
#include <string>
#include "tablefield.h"
```

Include dependency graph for tablerow.h:

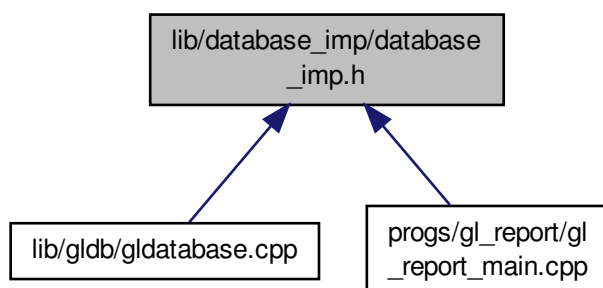




Include dependency graph for database\_imp.h:



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.*

### 9.15.1 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/>

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

Implementation of Dummy database connection implementation class.

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

```
graph TD; A["lib/database_imp/dummy/dbconn_dummy_imp.cpp"] --> B["sstream"]; A --> C["dbconn_dummy_imp.h"]; C --> D["database/database.h"]; D --> E["dbconn.h"]; D --> F["dbconnimp.h"]; D --> G["data_structures.h"]; D --> H["string"]; E --> I["memory"]; E --> F; E --> G; E --> H; F --> G; F --> H; G --> J["table.h"]; G --> K["tablerow.h"]; G --> L["tablefield.h"]; G --> H; J --> M["stdexcept"]; J --> N["vector"]; J --> K; J --> H; K --> N; K --> L; K --> H; L --> O["iostream"]; L --> H;
```

### Implementation of Dummy database connection implementation class.

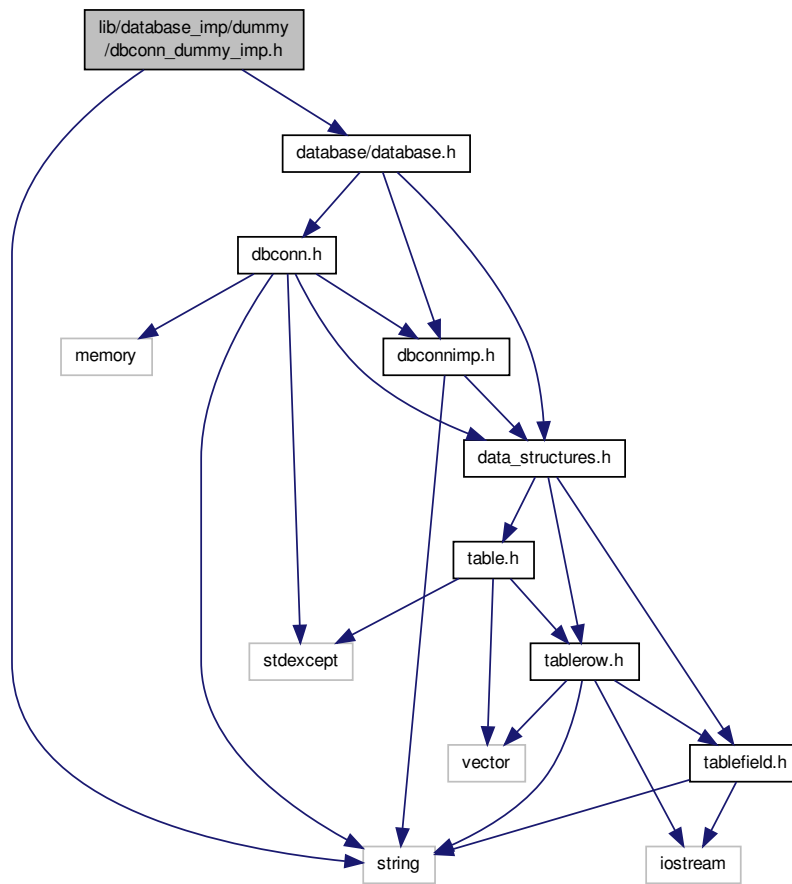
Paul Griffiths

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

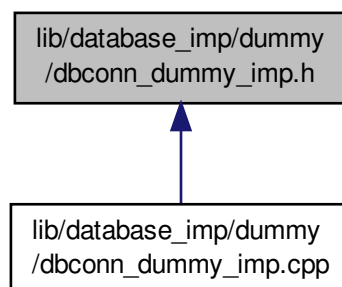
Interface to dummy database connection implementation class.

```
#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.*

### 9.17.1 Detailed Description

Interface to dummy database connection implementation class.

#### Author

Paul Griffiths

#### Copyright

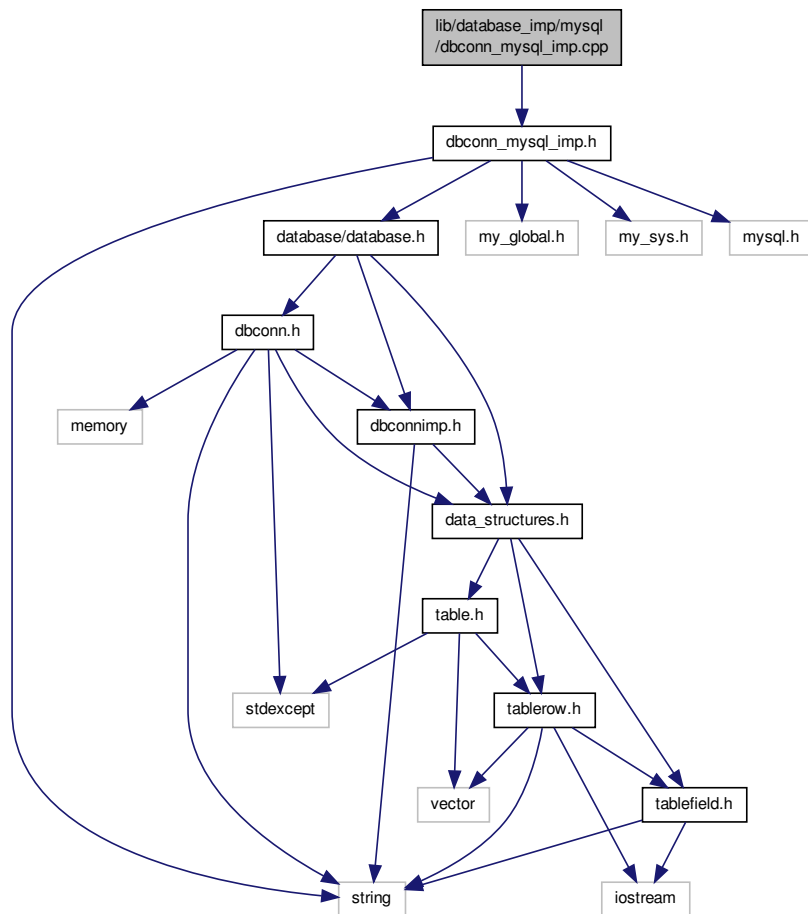
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 9.18 lib/database\_imp/mysql/dbconn\_mysql\_imp.cpp File Reference

Implementation of MySQL database connection implementation class.

```
#include "dbconn_mysql_imp.h"
```

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



### 9.18.1 Detailed Description

Implementation of MySQL database connection implementation class.

#### Author

Paul Griffiths

#### Copyright

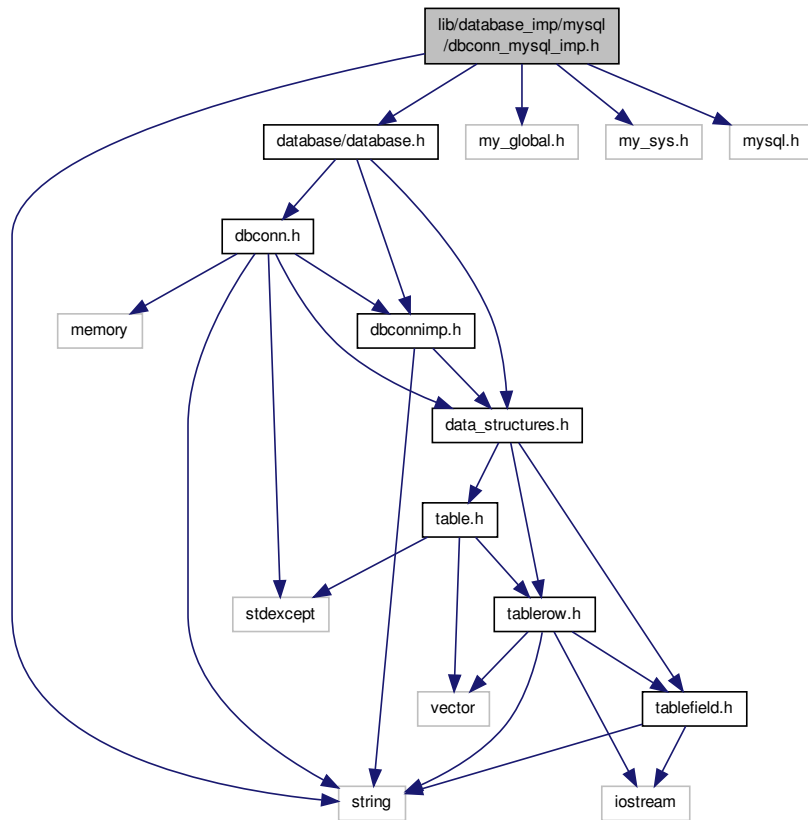
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

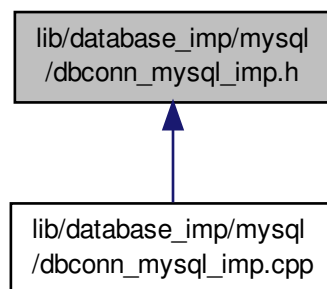
Interface to MySQL database connection implementation class.

```
#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.*

### 9.19.1 Detailed Description

Interface to MySQL database connection implementation class.

#### Author

Paul Griffiths

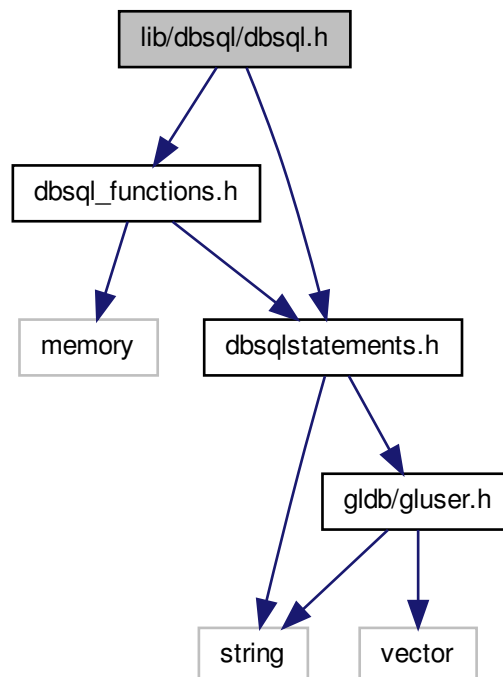
#### Copyright

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

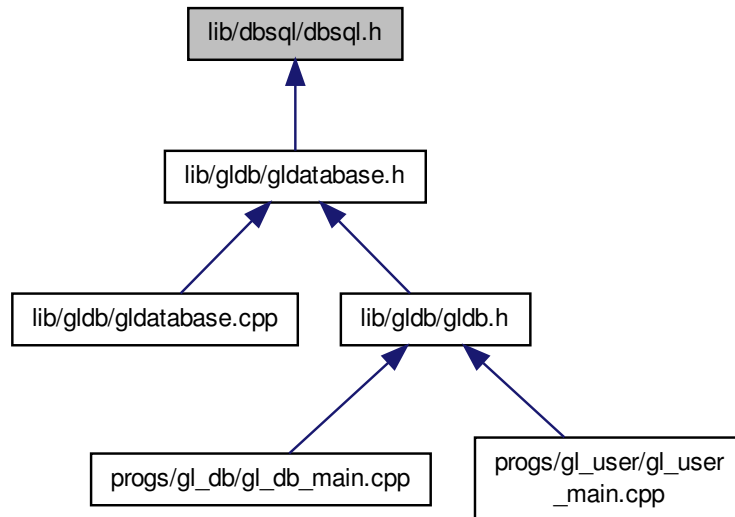
## 9.20 lib/dbsql/dbsql.h File Reference

User interface to DBSQL module.

```
#include "dbsql_functions.h"  
#include "dbsqlstatements.h"  
Include dependency graph for dbsql.h:
```



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



### 9.20.1 Detailed Description

User interface to DBSQL module.

#### Author

Paul Griffiths

#### Copyright

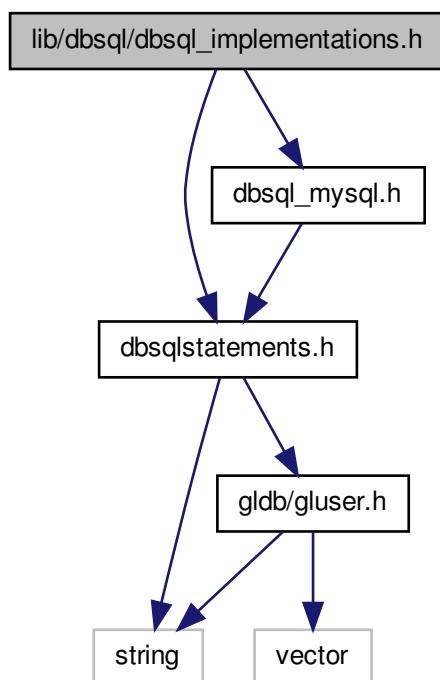
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 9.21 lib/dbsql/dbsql\_implementations.h File Reference

Aggregation header for DBSqlStatements implementations.

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

Include dependency graph for `dbsql_implementations.h`:



### 9.21.1 Detailed Description

Aggregation header for DBSqlStatements implementations.

#### Author

Paul Griffiths

## Copyright

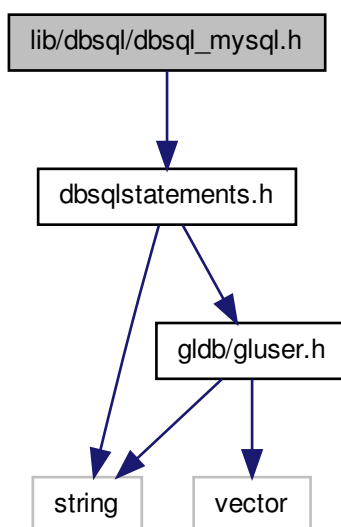
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

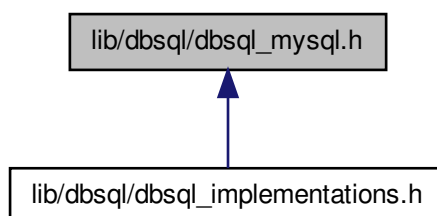
Interface to MySQL SQL statement class.

```
#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.*

### 9.22.1 Detailed Description

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/>

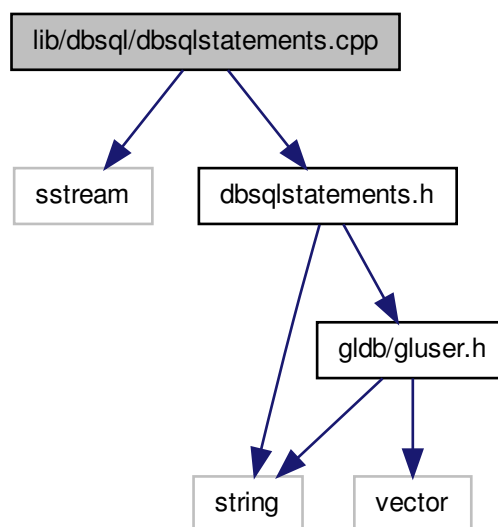
## 9.23 lib/dbsql/dbsqlstatements.cpp File Reference

Implementation of SQL statement class.

```
#include <sstream>
```

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

Include dependency graph for dbsqlstatements.cpp:



### 9.23.1 Detailed Description

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/>

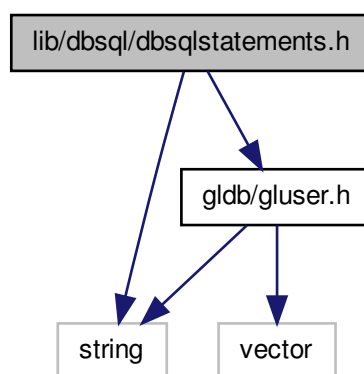
## 9.24 lib/dbsql/dbsqlstatements.h File Reference

Implementation of SQL module standalone functions.

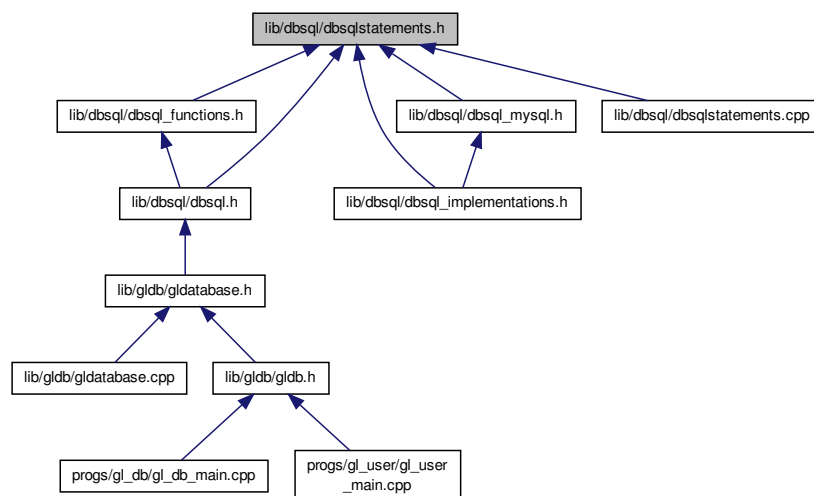
```
#include <string>
```

```
#include "gldb/gluser.h"
```

Include dependency graph for dbsqlstatements.h:



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



## Classes

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

### 9.24.1 Detailed Description

Implementation of SQL module standalone functions. Interface to SQL statements class.

Interface to SQL module standalone functions.

#### Author

Paul Griffiths

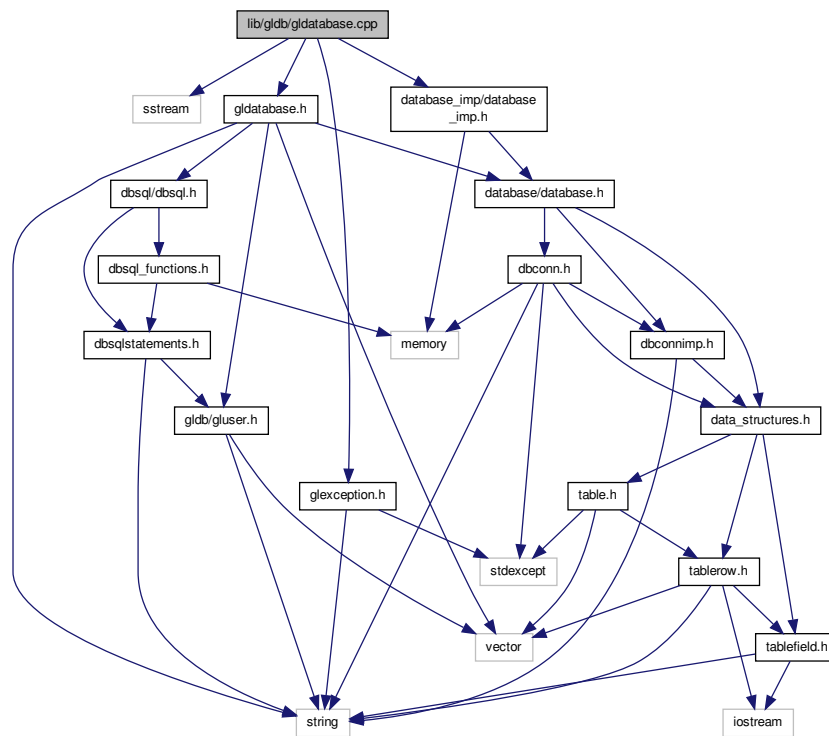
#### Copyright

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

## 9.25 lib/gldb/gldatabase.cpp File Reference

Implementation of General Ledger database class.

```
#include <sstream>
#include "gldatabase.h"
#include "glexception.h"
#include "database_imp/database_imp.h"
Include dependency graph for gldatabase.cpp:
```



## Functions

- static bool [boolstring\\_to\\_bool](#) (const std::string &bs)  
*Converts a string representation of a bool to a bool.*
- **m\_views** ({ "current\_trial\_balance", "check\_total", "all\_jes" })

### 9.25.1 Detailed Description

Implementation of General Ledger database class.

#### Author

Paul Griffiths

#### Copyright

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

### 9.25.2 Function Documentation

#### 9.25.2.1 static bool boolstring\_to\_bool ( const std::string & bs ) [static]

Converts a string representation of a bool to a bool.

#### Parameters

<i>bs</i>	The bool string.
-----------	------------------

#### Returns

true if *bs* contains "1" or "TRUE", false if *bs* contains "0" or "FALSE".

#### Exceptions

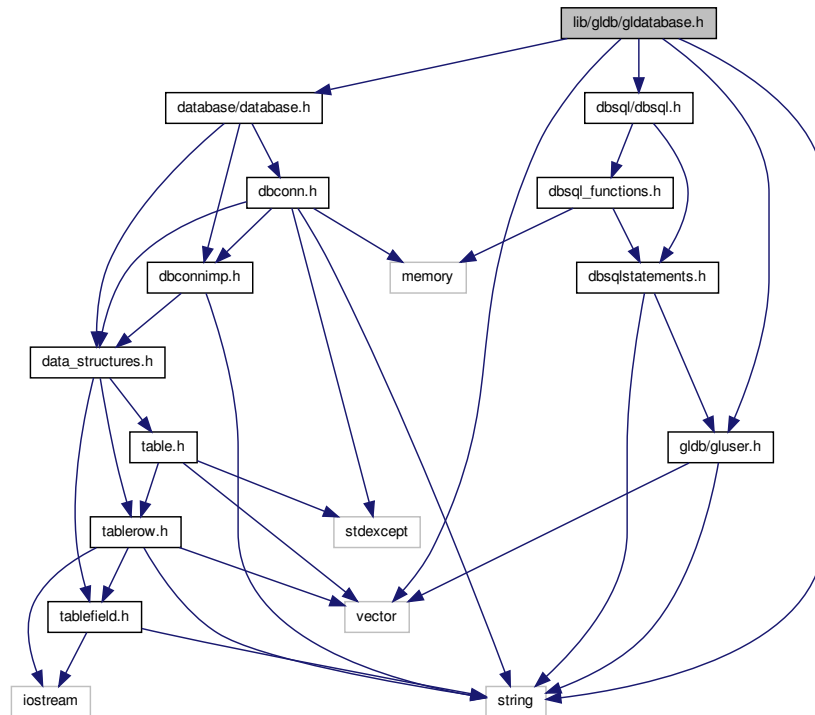
<i>GLDBException</i>	if <i>bs</i> contains any other value.
----------------------	--

## 9.26 lib/gldb/gldatabase.h File Reference

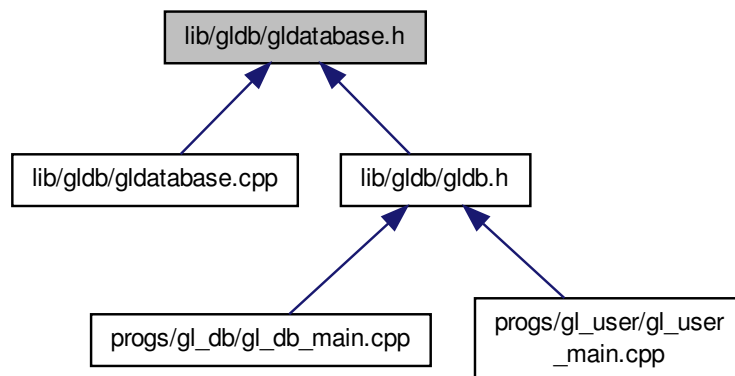
Interface to General Ledger database class.

```
#include <vector>
#include <string>
#include "database/database.h"
#include "dbsql/dbsql.h"
#include "gluser.h"
```

Include dependency graph for `gldatabase.h`:



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

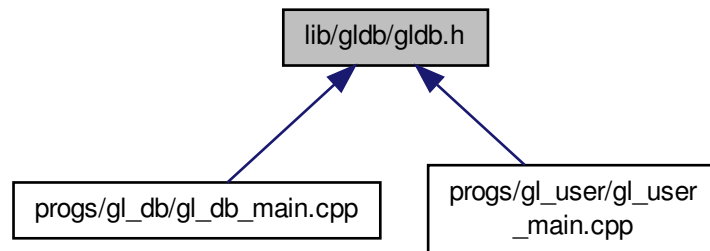


## Classes

- class [genleg::GLDatabase](#)  
*General ledger database class.*



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



### 9.27.1 Detailed Description

User interface to General Ledger database module.

#### Author

Paul Griffiths

#### Copyright

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

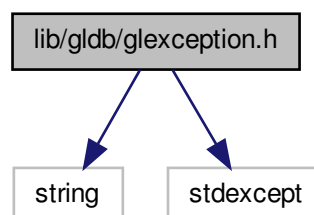
## 9.28 lib/gldb/glexception.h File Reference

Interface to General Ledger base exception class.

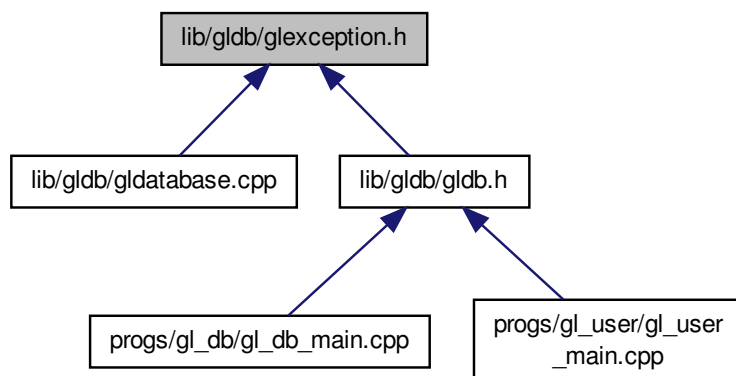
```
#include <string>
```

```
#include <stdexcept>
```

Include dependency graph for `glexception.h`:



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



## Classes

- class [genleg::GLDBException](#)

*Base general ledger database exception class.*

### 9.28.1 Detailed Description

Interface to General Ledger base exception class.

#### Author

Paul Griffiths

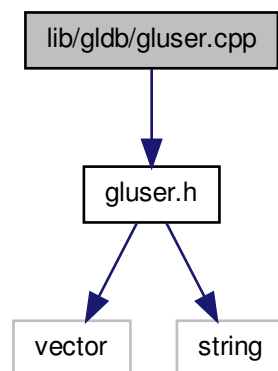
### Copyright

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

## 9.29 lib/gldb/gluser.cpp File Reference

Implementation of user class.

```
#include "gluser.h"
Include dependency graph for gluser.cpp:
```



### 9.29.1 Detailed Description

Implementation of user class.

#### Author

Paul Griffiths

### Copyright

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

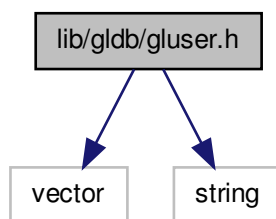
## 9.30 lib/gldb/gluser.h File Reference

Interface to user class.

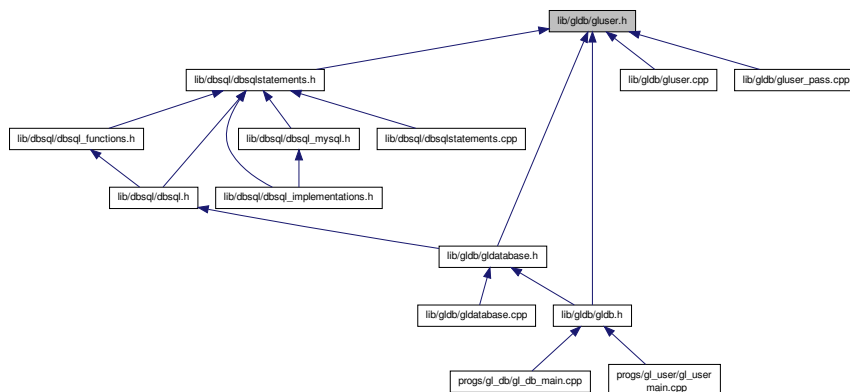
```
#include <vector>
#include <string>
```



Include dependency graph for gluser.h:



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



## Classes

- class [genleg::GLUser](#)  
*General ledger user class.*

### 9.30.1 Detailed Description

Interface to user class.

#### Author

Paul Griffiths

#### Copyright

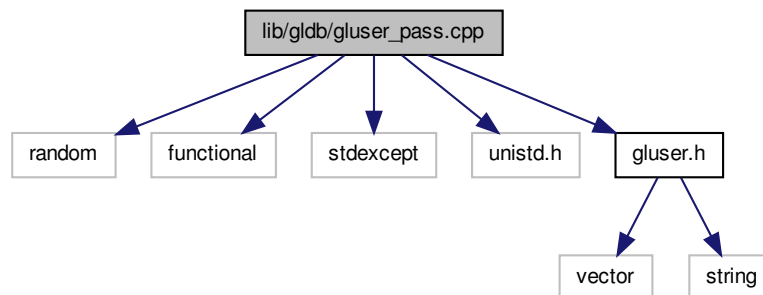
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 9.31 lib/gldb/gluser\_pass.cpp File Reference

Implementation of password functions for user class.

```
#include <random>
#include <functional>
#include <stdexcept>
#include <unistd.h>
#include "gluser.h"
```

Include dependency graph for gluser\_pass.cpp:



### Macros

- `#define _XOPEN_SOURCE 600`

### Functions

- static `std::string generate_salt ()`  
*Generates a random two-character salt for crypt()*

#### 9.31.1 Detailed Description

Implementation of password functions for user class.

**Todo** Implement a better form of password encryption. In particular, these functions are not re-entrant, and only use the first 8 characters of the password.

### Author

Paul Griffiths

### Copyright

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

## 9.31.2 Macro Definition Documentation

### 9.31.2.1 `#define _XOPEN_SOURCE 600`

UNIX feature test macro

## 9.31.3 Function Documentation

### 9.31.3.1 `static std::string generate_salt ( ) [static]`

Generates a random two-character salt for crypt()

#### Returns

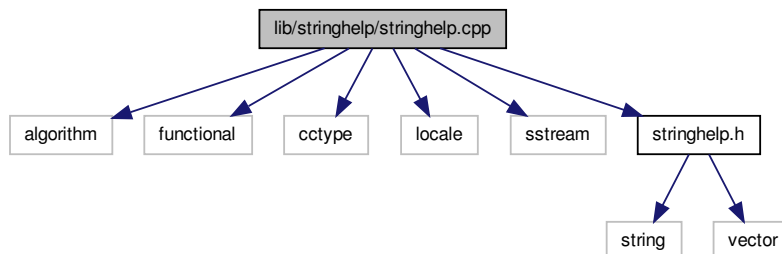
The two-character salt.

## 9.32 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:



### 9.32.1 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/>

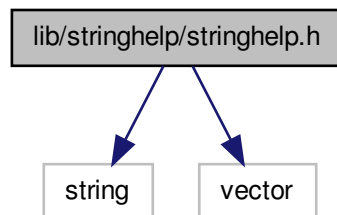
### 9.33 lib/stringhelp/stringhelp.h File Reference

Interface to string helper functions.

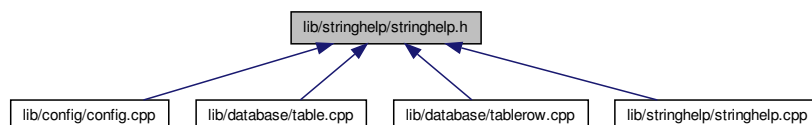
```
#include <string>
```

```
#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.*

- bool **pgstring::replace** (std::string &str, const std::string &from, const std::string &to)

*Replaces a substring with another string.*

### 9.33.1 Detailed Description

Interface to string helper functions.

**Author**

Paul Griffiths

**Copyright**

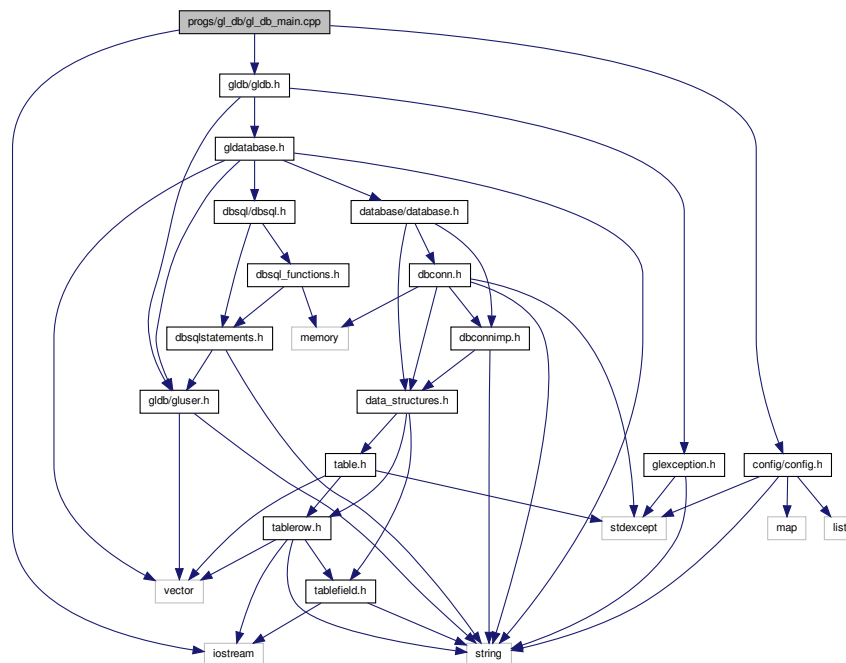
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

Main functionality for gl\_db program.

```
#include <iostream>
#include "gldb/gldb.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 bool `check_help_and_version` (const `Config` &config)  
*Prints help or version messages if requested.*
- static bool `check_db_parameters` (const `Config` &config)  
*Checks if database, hostname and username were provided.*
- 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 \* `programe` = "gl\_db"  
*Static variable for program name.*

### 9.34.1 Detailed Description

Main functionality for gl\_db program.

#### Author

Paul Griffiths

#### Copyright

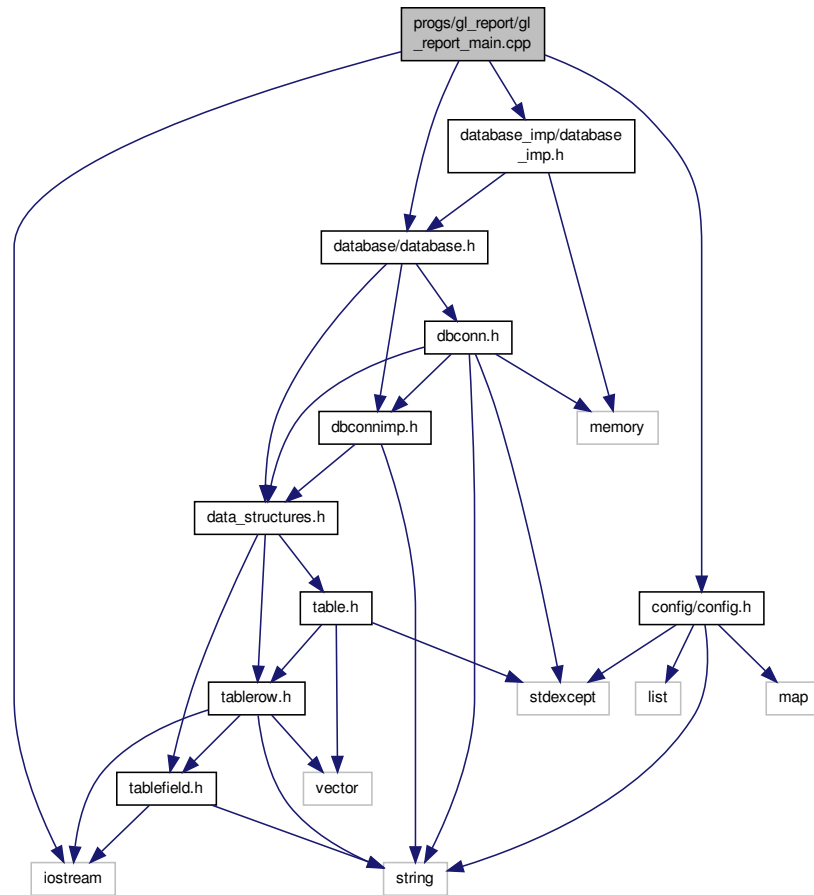
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 9.35 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.*





- `GLUser get_user (Config &config, GLDatabase &gdb)`  
*Returns a user from either an ID or a name.*
- `static void show_user_details (const GLUser &user)`  
*Outputs details for a user.*
- `static void enable_user (GLUser &user, Config &config, GLDatabase &gdb)`  
*Enables or disables a user.*
- `static void set_user_password (GLUser &user, Config &config, GLDatabase &gdb)`  
*Sets a user's password.*
- `static void check_user_password (GLUser &user, Config &config)`  
*Checks a user's password.*
- `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 * progname = "gl_user"`  
*Static variable for program name.*

### 9.36.1 Detailed Description

Main functionality for gl\_user program.

#### Author

Paul Griffiths

#### Copyright

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

### 9.36.2 Function Documentation

#### 9.36.2.1 `static bool check_db_parameters ( const Config & config ) [static]`

Checks if database, hostname and username were provided.

#### Parameters

<i>config</i>	Reference to a Config object.
---------------	-------------------------------

#### Returns

`true` if the information was provided, `false` otherwise.

### 9.36.2.2 static bool check\_help\_and\_version ( const Config & config ) [static]

Prints help or version messages if requested.

#### Parameters

<i>config</i>	Reference to a Config object.
---------------	-------------------------------

#### Returns

`true` if the help or version message was requested, `false` otherwise.

### 9.36.2.3 static void check\_user\_password ( GLUser & user, Config & config ) [static]

Checks a user's password.

#### Parameters

<i>user</i>	Reference to user.
<i>config</i>	Reference to program configuration options.

### 9.36.2.4 static void enable\_user ( GLUser & user, Config & config, GLDatabase & gdb ) [static]

Enables or disables a user.

#### Parameters

<i>user</i>	Reference to user.
<i>config</i>	Reference to program configuration.
<i>gdb</i>	Reference to database object.

### 9.36.2.5 GLUser get\_user ( Config & config, GLDatabase & gdb )

Returns a user from either an ID or a name.

#### Parameters

<i>config</i>	Program configurations object.
<i>gdb</i>	Database object.

#### Returns

The user.

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

Gets a password from the terminal.

#### Returns

The password.

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

Main function.

#### Parameters

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

#### Returns

Exit status code.

### 9.36.2.8 `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> .

### 9.36.2.9 `static void set_user_password ( GLUser & user, Config & config, GLDatabase & gdb )` `[static]`

Sets a user's password.

#### Parameters

<i>user</i>	Reference to user.
<i>config</i>	Reference to program configuration.
<i>gdb</i>	Reference to database object.

### 9.36.2.10 `static void show_user_details ( const GLUser & user )` `[static]`

Outputs details for a user.

#### Parameters

<i>user</i>	Reference to user.
-------------	--------------------

# Index

- ~Config
  - genleg::Config, [25](#)
- ~DBConnDummy
  - gldb::DBConnDummy, [39](#)
- ~DBConnImp
  - gldb::DBConnImp, [41](#)
- ~DBConnMySQL
  - gldb::DBConnMySQL, [43](#)
- ~DBSQLStatements
  - genleg::DBSQLStatements, [46](#)
- ~GLDatabase
  - genleg::GLDatabase, [51](#)
- ~GLUser
  - genleg::GLUser, [56](#)
- ~Table
  - gldb::Table, [61](#)
- ~TableField
  - gldb::TableField, [68](#)
- ~TableRow
  - gldb::TableRow, [75](#)
- \_XOPEN\_SOURCE
  - config\_getopt.cpp, [82](#)
  - gluser\_pass.cpp, [119](#)
- add\_cmdline\_option
  - genleg::Config, [26](#)
- append\_field
  - gldb::TableRow, [75](#)
- append\_record
  - gldb::Table, [61](#)
- backend
  - genleg::GLDatabase, [51](#)
- boolstring\_to\_bool
  - gldatabase.cpp, [111](#)
- check\_db\_parameters
  - Database program., [22](#)
  - gl\_user\_main.cpp, [125](#)
- check\_help\_and\_version
  - Database program., [22](#)
  - gl\_user\_main.cpp, [125](#)
- check\_password
  - genleg::GLUser, [56](#)
- check\_user\_password
  - gl\_user\_main.cpp, [126](#)
- Config
  - genleg::Config, [25](#)
- config\_getopt.cpp
  - \_XOPEN\_SOURCE, [82](#)
- ConfigBadConfigFile
  - genleg::ConfigBadConfigFile, [28](#)
- ConfigBadOption
  - genleg::ConfigBadOption, [29](#)
- ConfigCouldNotOpenFile
  - genleg::ConfigCouldNotOpenFile, [31](#)
- ConfigException
  - genleg::ConfigException, [32](#)
- ConfigOptionNotSet
  - genleg::ConfigOptionNotSet, [33](#)
- create\_from\_file
  - gldb::Table, [61](#)
- create\_structure
  - genleg::GLDatabase, [51](#)
- create\_table
  - genleg::DBSQLStatements, [47](#)
- create\_user
  - genleg::GLDatabase, [52](#)
- create\_view
  - genleg::DBSQLStatements, [47](#)
- DBConn
  - gldb::DBConn, [34](#)
- DBConnCouldNotConnect
  - gldb::DBConnCouldNotConnect, [36](#)
- DBConnCouldNotQuery
  - gldb::DBConnCouldNotQuery, [37](#)
- DBConnDummy
  - gldb::DBConnDummy, [38, 39](#)
- DBConnException
  - gldb::DBConnException, [40](#)
- DBConnImp
  - gldb::DBConnImp, [41](#)
- DBConnMySQL
  - gldb::DBConnMySQL, [43](#)
- DBSQLStatements
  - genleg::DBSQLStatements, [46](#)
- Database interaction module, [14](#)
- get\_connection, [15](#)
- get\_database\_type, [15](#)
- Database program., [22](#)
  - check\_db\_parameters, [22](#)
  - check\_help\_and\_version, [22](#)
  - login, [23](#)
  - main, [23](#)
  - set\_configuration, [23](#)
- destroy\_structure
  - genleg::GLDatabase, [52](#)
- drop\_table
  - genleg::DBSQLStatements, [47](#)

- drop\_view
  - genleg::DBSQLStatements, 47
- enable\_user
  - gl\_user\_main.cpp, 126
- enabled
  - genleg::GLUser, 57
- firstname
  - genleg::GLUser, 57
- GLDBException
  - genleg::GLDBException, 54
- GLDatabase
  - genleg::GLDatabase, 51
- GLUser
  - genleg::GLUser, 56
- General Ledger database module., 13
- General purpose helpers., 18
  - split, 18
  - trim, 18
  - trim\_back, 19
  - trim\_front, 19
- generate\_salt
  - gluser\_pass.cpp, 119
- genleg::Config, 25
  - ~Config, 25
  - add\_cmdline\_option, 26
  - Config, 25
  - is\_set, 26
  - m\_opts\_set, 27
  - m\_opts\_supp, 27
  - populate\_from\_cmdline, 26
  - populate\_from\_file, 27
- genleg::ConfigBadConfigFile, 27
  - ConfigBadConfigFile, 28
- genleg::ConfigBadOption, 29
  - ConfigBadOption, 29
- genleg::ConfigCouldNotOpenFile, 30
  - ConfigCouldNotOpenFile, 31
- genleg::ConfigException, 31
  - ConfigException, 32
- genleg::ConfigOptionNotSet, 32
  - ConfigOptionNotSet, 33
- genleg::DBSQLMySQL, 44
- genleg::DBSQLStatements, 45
  - ~DBSQLStatements, 46
  - create\_table, 47
  - create\_view, 47
  - DBSQLStatements, 46
  - drop\_table, 47
  - drop\_view, 47
  - get\_perms, 47
  - grant, 48
  - revoke, 48
  - update\_user, 48
  - user\_by\_id, 49
  - user\_by\_username, 49
- genleg::GLDBException, 54
  - GLDBException, 54
- genleg::GLDatabase, 49
  - ~GLDatabase, 51
  - backend, 51
  - create\_structure, 51
  - create\_user, 52
  - destroy\_structure, 52
  - GLDatabase, 51
  - get\_user\_by\_id, 52
  - get\_user\_by\_username, 52
  - grant, 53
  - load\_sample\_data, 53
  - m\_dbc, 53
  - m\_sql, 54
  - m\_tables, 54
  - m\_views, 54
  - revoke, 53
  - update\_user, 53
- genleg::GLUser, 55
  - ~GLUser, 56
  - check\_password, 56
  - enabled, 57
  - firstname, 57
  - GLUser, 56
  - id, 57
  - lastname, 57
  - m\_enabled, 59
  - m\_firstname, 59
  - m\_id, 59
  - m\_lastname, 59
  - m\_pass\_hash, 59
  - m\_pass\_salt, 59
  - m\_perms, 59
  - m\_username, 59
  - pass\_hash, 57
  - pass\_salt, 57
  - permissions, 58
  - set\_enabled, 58
  - set\_firstname, 58
  - set\_lastname, 58
  - set\_password, 58
  - set\_username, 58
  - username, 59
- get\_connection
  - Database interaction module, 15
- get\_database\_type
  - Database interaction module, 15
- get\_field
  - gldb::Table, 61
- get\_headers
  - gldb::Table, 62
- get\_perms
  - genleg::DBSQLStatements, 47
- get\_user
  - gl\_user\_main.cpp, 126
- get\_user\_by\_id
  - genleg::GLDatabase, 52
- get\_user\_by\_username

- genleg::GLDatabase, 52
- gl\_user\_main.cpp
  - check\_db\_parameters, 125
  - check\_help\_and\_version, 125
  - check\_user\_password, 126
  - enable\_user, 126
  - get\_user, 126
  - login, 126
  - main, 126
  - set\_configuration, 127
  - set\_user\_password, 127
  - show\_user\_details, 127
- gldatabase.cpp
  - boolstring\_to\_bool, 111
- gldb::DBConn, 33
  - DBConn, 34
  - m\_imp, 35
  - operator=, 34
  - query, 34
  - select, 34
- gldb::DBConnCouldNotConnect, 35
  - DBConnCouldNotConnect, 36
- gldb::DBConnCouldNotQuery, 36
  - DBConnCouldNotQuery, 37
- gldb::DBConnDummy, 37
  - ~DBConnDummy, 39
  - DBConnDummy, 38, 39
  - operator=, 39
  - select, 39
- gldb::DBConnException, 39
  - DBConnException, 40
- gldb::DBConnImp, 40
  - ~DBConnImp, 41
  - DBConnImp, 41
  - query, 41
  - select, 41
- gldb::DBConnMySQL, 42
  - ~DBConnMySQL, 43
  - DBConnMySQL, 43
  - m\_conn, 44
  - operator=, 43
  - query, 44
  - select, 44
- gldb::Table, 60
  - ~Table, 61
  - append\_record, 61
  - create\_from\_file, 61
  - get\_field, 61
  - get\_headers, 62
  - insert\_query, 62
  - m\_headers, 63
  - m\_quoted, 63
  - m\_records, 63
  - num\_fields, 62
  - num\_records, 62
  - set\_quoted, 63
  - Table, 61
- gldb::TableBadInputFile, 63
  - TableBadInputFile, 64
- gldb::TableCouldNotOpenInputFile, 65
  - TableCouldNotOpenInputFile, 65
- gldb::TableException, 66
  - TableException, 66
- gldb::TableField, 67
  - ~TableField, 68
  - length, 68
  - m\_data, 70
  - operator std::string, 68
  - operator<<, 70
  - operator+=, 69
  - operator=, 69
  - TableField, 68
- gldb::TableMismatchedRecordLength, 70
  - TableMismatchedRecordLength, 71
- gldb::TableNoSuchField, 72
  - TableNoSuchField, 73
- gldb::TableNoSuchRecord, 73
  - TableNoSuchRecord, 74
- gldb::TableRow, 74
  - ~TableRow, 75
  - append\_field, 75
  - m\_fields, 77
  - print, 76
  - record\_string, 76
  - size, 77
  - TableRow, 75
- gluser\_pass.cpp
  - \_XOPEN\_SOURCE, 119
  - generate\_salt, 119
- grant
  - genleg::DBSQLStatements, 48
  - genleg::GLDatabase, 53
- id
  - genleg::GLUser, 57
- insert\_query
  - gldb::Table, 62
- is\_set
  - genleg::Config, 26
- lastname
  - genleg::GLUser, 57
- length
  - gldb::TableField, 68
- lib/config/config.cpp, 79
- lib/config/config.h, 80
- lib/config/config\_getopt.cpp, 81
- lib/database/data\_structures.h, 82
- lib/database/database.h, 83
- lib/database/dbconn.cpp, 85
- lib/database/dbconn.h, 86
- lib/database/dbconnimp.h, 87
- lib/database/table.cpp, 89
- lib/database/table.h, 90
- lib/database/tablefield.cpp, 92
- lib/database/tablefield.h, 92
- lib/database/tablerow.cpp, 94

- lib/database/tablerow.h, 95
- lib/database\_imp/database\_imp.h, 96
- lib/database\_imp/dummy/dbconn\_dummy\_imp.cpp, 98
- lib/database\_imp/dummy/dbconn\_dummy\_imp.h, 99
- lib/database\_imp/mysql/dbconn\_mysql\_imp.cpp, 101
- lib/database\_imp/mysql/dbconn\_mysql\_imp.h, 102
- lib/dbsql/dbsql.h, 104
- lib/dbsql/dbsql\_implementations.h, 105
- lib/dbsql/dbsql\_mysql.h, 107
- lib/dbsql/dbsqlstatements.cpp, 108
- lib/dbsql/dbsqlstatements.h, 109
- lib/gldb/gldatabase.cpp, 110
- lib/gldb/gldatabase.h, 111
- lib/gldb/gldb.h, 113
- lib/gldb/glexception.h, 114
- lib/gldb/gluser.cpp, 116
- lib/gldb/gluser.h, 116
- lib/gldb/gluser\_pass.cpp, 118
- lib/stringhelp/stringhelp.cpp, 119
- lib/stringhelp/stringhelp.h, 120
- load\_sample\_data
  - genleg::GLDatabase, 53
- login
  - Database program., 23
  - gl\_user\_main.cpp, 126
  - Reporting program., 20
- m\_conn
  - gldb::DBConnMySQL, 44
- m\_data
  - gldb::TableField, 70
- m\_dbc
  - genleg::GLDatabase, 53
- m\_enabled
  - genleg::GLUser, 59
- m\_fields
  - gldb::TableRow, 77
- m\_firstname
  - genleg::GLUser, 59
- m\_headers
  - gldb::Table, 63
- m\_id
  - genleg::GLUser, 59
- m\_imp
  - gldb::DBConn, 35
- m\_lastname
  - genleg::GLUser, 59
- m\_opts\_set
  - genleg::Config, 27
- m\_opts\_supp
  - genleg::Config, 27
- m\_pass\_hash
  - genleg::GLUser, 59
- m\_pass\_salt
  - genleg::GLUser, 59
- m\_perms
  - genleg::GLUser, 59
- m\_quoted
  - gldb::Table, 63
- m\_records
  - gldb::Table, 63
- m\_sql
  - genleg::GLDatabase, 54
- m\_tables
  - genleg::GLDatabase, 54
- m\_username
  - genleg::GLUser, 59
- m\_views
  - genleg::GLDatabase, 54
- main
  - Database program., 23
  - gl\_user\_main.cpp, 126
  - Reporting program., 20
- num\_fields
  - gldb::Table, 62
- num\_records
  - gldb::Table, 62
- operator std::string
  - gldb::TableField, 68
- operator<<
  - gldb::TableField, 70
- operator+=
  - gldb::TableField, 69
- operator=
  - gldb::DBConn, 34
  - gldb::DBConnDummy, 39
  - gldb::DBConnMySQL, 43
  - gldb::TableField, 69
- pass\_hash
  - genleg::GLUser, 57
- pass\_salt
  - genleg::GLUser, 57
- permissions
  - genleg::GLUser, 58
- populate\_from\_cmdline
  - genleg::Config, 26
- populate\_from\_file
  - genleg::Config, 27
- print
  - gldb::TableRow, 76
- Program configuration module, 17
- progs/gl\_db/gl\_db\_main.cpp, 121
- progs/gl\_report/gl\_report\_main.cpp, 122
- progs/gl\_user/gl\_user\_main.cpp, 124
- query
  - gldb::DBConn, 34
  - gldb::DBConnImp, 41
  - gldb::DBConnMySQL, 44
- record\_string
  - gldb::TableRow, 76
- Reporting program., 20
  - login, 20
  - main, 20

- set\_configuration, 21
- revoke
  - genleg::DBSQLStatements, 48
  - genleg::GLDatabase, 53
- SQL statements module, 16
- select
  - gldb::DBConn, 34
  - gldb::DBConnDummy, 39
  - gldb::DBConnImp, 41
  - gldb::DBConnMySQL, 44
- set\_configuration
  - Database program., 23
  - gl\_user\_main.cpp, 127
  - Reporting program., 21
- set\_enabled
  - genleg::GLUser, 58
- set\_firstname
  - genleg::GLUser, 58
- set\_lastname
  - genleg::GLUser, 58
- set\_password
  - genleg::GLUser, 58
- set\_quoted
  - gldb::Table, 63
- set\_user\_password
  - gl\_user\_main.cpp, 127
- set\_username
  - genleg::GLUser, 58
- show\_user\_details
  - gl\_user\_main.cpp, 127
- size
  - gldb::TableRow, 77
- split
  - General purpose helpers., 18
- Table
  - gldb::Table, 61
- TableBadInputFile
  - gldb::TableBadInputFile, 64
- TableCouldNotOpenInputFile
  - gldb::TableCouldNotOpenInputFile, 65
- TableException
  - gldb::TableException, 66
- TableField
  - gldb::TableField, 68
- TableMismatchedRecordLength
  - gldb::TableMismatchedRecordLength, 71
- TableNoSuchField
  - gldb::TableNoSuchField, 73
- TableNoSuchRecord
  - gldb::TableNoSuchRecord, 74
- TableRow
  - gldb::TableRow, 75
- trim
  - General purpose helpers., 18
- trim\_back
  - General purpose helpers., 19
- trim\_front
  - General purpose helpers., 19
- update\_user
  - genleg::DBSQLStatements, 48
  - genleg::GLDatabase, 53
- user\_by\_id
  - genleg::DBSQLStatements, 49
- user\_by\_username
  - genleg::DBSQLStatements, 49
- username
  - genleg::GLUser, 59