

general_ledger

Generated by Doxygen 1.8.1.2

Mon Jun 16 2014 19:24:24

Contents

1	General Ledger.	1
2	Todo List	3
3	Bug List	5
4	Module Index	7
4.1	Modules	7
5	Class Index	9
5.1	Class Hierarchy	9
6	Class Index	11
6.1	Class List	11
7	File Index	13
7.1	File List	13
8	Module Documentation	15
8.1	General Ledger database module.	15
8.1.1	Detailed Description	15
8.2	Database interaction module	16
8.2.1	Detailed Description	17
8.2.2	Function Documentation	17
8.2.2.1	get_connection	17
8.2.2.2	get_database_type	17
8.2.2.3	get_field_names	17
8.2.2.4	get_row	17
8.3	SQL statements module	19
8.3.1	Detailed Description	19
8.4	Program configuration module	20
8.4.1	Detailed Description	20
8.5	General purpose helpers.	21
8.5.1	Detailed Description	21

8.5.2	Function Documentation	21
8.5.2.1	split	21
8.5.2.2	split	21
8.5.2.3	trim	21
8.5.2.4	trim_back	22
8.5.2.5	trim_front	22
8.6	Reporting program.	23
8.6.1	Detailed Description	23
8.6.2	Function Documentation	23
8.6.2.1	login	23
8.6.2.2	main	23
8.6.2.3	set_configuration	24
8.7	Database program.	25
8.7.1	Detailed Description	25
8.7.2	Function Documentation	25
8.7.2.1	check_db_parameters	25
8.7.2.2	check_help_and_version	25
8.7.2.3	login	26
8.7.2.4	main	26
8.7.2.5	set_configuration	26
9	Class Documentation	27
9.1	genleg::Config Class Reference	27
9.1.1	Detailed Description	27
9.1.2	Constructor & Destructor Documentation	27
9.1.2.1	Config	27
9.1.2.2	~Config	28
9.1.3	Member Function Documentation	28
9.1.3.1	add_cmdline_option	28
9.1.3.2	is_set	28
9.1.3.3	operator[]	28
9.1.3.4	populate_from_cmdline	28
9.1.3.5	populate_from_file	29
9.1.4	Member Data Documentation	29
9.1.4.1	m_opts_set	29
9.1.4.2	m_opts_supp	29
9.2	genleg::ConfigBadConfigFile Class Reference	29
9.2.1	Detailed Description	30
9.2.2	Constructor & Destructor Documentation	30
9.2.2.1	ConfigBadConfigFile	30

9.3	genleg::ConfigBadOption Class Reference	31
9.3.1	Detailed Description	31
9.3.2	Constructor & Destructor Documentation	31
9.3.2.1	ConfigBadOption	32
9.4	genleg::ConfigCouldNotOpenFile Class Reference	32
9.4.1	Detailed Description	33
9.4.2	Constructor & Destructor Documentation	33
9.4.2.1	ConfigCouldNotOpenFile	33
9.5	genleg::ConfigException Class Reference	33
9.5.1	Detailed Description	33
9.5.2	Constructor & Destructor Documentation	34
9.5.2.1	ConfigException	34
9.6	genleg::ConfigOptionNotSet Class Reference	34
9.6.1	Detailed Description	35
9.6.2	Constructor & Destructor Documentation	35
9.6.2.1	ConfigOptionNotSet	35
9.7	gldb::DBConn Class Reference	35
9.7.1	Detailed Description	36
9.7.2	Constructor & Destructor Documentation	36
9.7.2.1	DBConn	36
9.7.2.2	DBConn	36
9.7.2.3	DBConn	36
9.7.3	Member Function Documentation	36
9.7.3.1	operator=	36
9.7.3.2	operator=	36
9.7.3.3	query	36
9.7.3.4	select	37
9.7.4	Member Data Documentation	37
9.7.4.1	m_imp	37
9.8	gldb::DBConnCouldNotConnect Class Reference	37
9.8.1	Detailed Description	38
9.8.2	Constructor & Destructor Documentation	38
9.8.2.1	DBConnCouldNotConnect	38
9.9	gldb::DBConnCouldNotQuery Class Reference	38
9.9.1	Detailed Description	39
9.9.2	Constructor & Destructor Documentation	39
9.9.2.1	DBConnCouldNotQuery	39
9.10	gldb::DBConnDummy Class Reference	40
9.10.1	Detailed Description	41
9.10.2	Constructor & Destructor Documentation	41

9.10.2.1	DBConnDummy	41
9.10.2.2	DBConnDummy	41
9.10.2.3	~DBConnDummy	41
9.10.3	Member Function Documentation	41
9.10.3.1	operator=	41
9.10.3.2	query	41
9.10.3.3	select	41
9.11	gldb::DBConnException Class Reference	42
9.11.1	Detailed Description	42
9.11.2	Constructor & Destructor Documentation	42
9.11.2.1	DBConnException	42
9.12	gldb::DBConnImp Class Reference	43
9.12.1	Detailed Description	43
9.12.2	Constructor & Destructor Documentation	43
9.12.2.1	DBConnImp	43
9.12.2.2	~DBConnImp	43
9.12.3	Member Function Documentation	44
9.12.3.1	query	44
9.12.3.2	select	44
9.13	gldb::DBConnMySQL Class Reference	44
9.13.1	Detailed Description	45
9.13.2	Constructor & Destructor Documentation	45
9.13.2.1	DBConnMySQL	45
9.13.2.2	DBConnMySQL	46
9.13.2.3	DBConnMySQL	46
9.13.2.4	~DBConnMySQL	46
9.13.3	Member Function Documentation	46
9.13.3.1	operator=	46
9.13.3.2	operator=	46
9.13.3.3	query	46
9.13.3.4	select	46
9.13.4	Member Data Documentation	47
9.13.4.1	m_conn	47
9.14	genleg::DBSQLDummy Class Reference	47
9.14.1	Detailed Description	48
9.15	genleg::DBSQLMySQL Class Reference	48
9.15.1	Detailed Description	48
9.16	genleg::DBSQLStatements Class Reference	49
9.16.1	Detailed Description	50
9.16.2	Constructor & Destructor Documentation	50

9.16.2.1	DBSQLStatements	50
9.16.2.2	~DBSQLStatements	50
9.16.3	Member Function Documentation	50
9.16.3.1	create_table	50
9.16.3.2	create_view	50
9.16.3.3	drop_table	50
9.16.3.4	drop_view	51
9.16.3.5	get_perms	51
9.16.3.6	grant	51
9.16.3.7	revoke	51
9.16.3.8	update_user	52
9.16.3.9	user_by_id	52
9.16.3.10	user_by_username	52
9.17	genleg::GLDatabase Class Reference	52
9.17.1	Detailed Description	54
9.17.2	Constructor & Destructor Documentation	54
9.17.2.1	GLDatabase	54
9.17.2.2	~GLDatabase	54
9.17.3	Member Function Documentation	54
9.17.3.1	backend	54
9.17.3.2	create_structure	55
9.17.3.3	create_user	55
9.17.3.4	destroy_structure	55
9.17.3.5	get_user_by_id	55
9.17.3.6	get_user_by_username	55
9.17.3.7	grant	56
9.17.3.8	load_sample_data	56
9.17.3.9	revoke	56
9.17.3.10	update_user	56
9.17.4	Member Data Documentation	56
9.17.4.1	m_dbc	57
9.17.4.2	m_sql	57
9.17.4.3	m_tables	57
9.17.4.4	m_views	57
9.18	genleg::GLDBException Class Reference	57
9.18.1	Detailed Description	57
9.18.2	Constructor & Destructor Documentation	57
9.18.2.1	GLDBException	57
9.19	genleg::GLUser Class Reference	58
9.19.1	Detailed Description	59

9.19.2	Constructor & Destructor Documentation	59
9.19.2.1	GLUser	59
9.19.2.2	~GLUser	59
9.19.3	Member Function Documentation	59
9.19.3.1	check_password	60
9.19.3.2	enabled	60
9.19.3.3	firstname	60
9.19.3.4	id	60
9.19.3.5	lastname	60
9.19.3.6	pass_hash	60
9.19.3.7	pass_salt	61
9.19.3.8	permissions	61
9.19.3.9	set_enabled	61
9.19.3.10	set_firstname	61
9.19.3.11	set_lastname	61
9.19.3.12	set_password	61
9.19.3.13	set_username	61
9.19.3.14	username	62
9.19.4	Member Data Documentation	62
9.19.4.1	m_enabled	62
9.19.4.2	m_firstname	62
9.19.4.3	m_id	62
9.19.4.4	m_lastname	62
9.19.4.5	m_pass_hash	62
9.19.4.6	m_pass_salt	62
9.19.4.7	m_perms	62
9.19.4.8	m_username	62
9.20	gldb::MySQLResult Class Reference	63
9.20.1	Detailed Description	63
9.20.2	Constructor & Destructor Documentation	63
9.20.2.1	MySQLResult	63
9.20.2.2	~MySQLResult	63
9.20.2.3	MySQLResult	63
9.20.2.4	MySQLResult	64
9.20.3	Member Function Documentation	64
9.20.3.1	num_fields	64
9.20.3.2	operator=	64
9.20.3.3	operator=	64
9.20.3.4	result	64
9.20.4	Member Data Documentation	64

9.20.4.1	m_num_fields	64
9.20.4.2	m_result	64
9.21	gldb::Table Class Reference	64
9.21.1	Detailed Description	66
9.21.2	Constructor & Destructor Documentation	66
9.21.2.1	Table	66
9.21.2.2	Table	66
9.21.2.3	Table	66
9.21.2.4	Table	66
9.21.2.5	~Table	67
9.21.3	Member Function Documentation	67
9.21.3.1	append_record	67
9.21.3.2	append_record	67
9.21.3.3	create_from_file	67
9.21.3.4	get_field	67
9.21.3.5	get_headers	68
9.21.3.6	insert_query	68
9.21.3.7	num_fields	68
9.21.3.8	num_records	68
9.21.3.9	operator=	68
9.21.3.10	operator=	69
9.21.3.11	operator[]	69
9.21.3.12	set_quoted	69
9.21.3.13	set_quoted	69
9.21.4	Member Data Documentation	69
9.21.4.1	m_headers	69
9.21.4.2	m_quoted	70
9.21.4.3	m_records	70
9.22	gldb::TableBadInputFile Class Reference	70
9.22.1	Detailed Description	71
9.22.2	Constructor & Destructor Documentation	71
9.22.2.1	TableBadInputFile	71
9.23	gldb::TableCouldNotOpenInputFile Class Reference	71
9.23.1	Detailed Description	72
9.23.2	Constructor & Destructor Documentation	72
9.23.2.1	TableCouldNotOpenInputFile	72
9.24	gldb::TableException Class Reference	72
9.24.1	Detailed Description	73
9.24.2	Constructor & Destructor Documentation	73
9.24.2.1	TableException	73

9.25	<code>gldb::TableField</code> Class Reference	73
9.25.1	Detailed Description	75
9.25.2	Constructor & Destructor Documentation	75
9.25.2.1	<code>TableField</code>	75
9.25.2.2	<code>TableField</code>	75
9.25.2.3	<code>TableField</code>	75
9.25.2.4	<code>TableField</code>	75
9.25.2.5	<code>TableField</code>	75
9.25.2.6	<code>~TableField</code>	76
9.25.3	Member Function Documentation	76
9.25.3.1	<code>length</code>	76
9.25.3.2	<code>operator std::string</code>	76
9.25.3.3	<code>operator+=</code>	76
9.25.3.4	<code>operator+=</code>	76
9.25.3.5	<code>operator=</code>	76
9.25.3.6	<code>operator=</code>	77
9.25.3.7	<code>operator=</code>	77
9.25.3.8	<code>operator=</code>	77
9.25.3.9	<code>operator=</code>	77
9.25.3.10	<code>operator[]</code>	78
9.25.3.11	<code>operator[]</code>	78
9.25.4	Friends And Related Function Documentation	78
9.25.4.1	<code>operator<<</code>	78
9.25.5	Member Data Documentation	78
9.25.5.1	<code>m_data</code>	78
9.26	<code>gldb::TableMismatchedRecordLength</code> Class Reference	79
9.26.1	Detailed Description	79
9.26.2	Constructor & Destructor Documentation	79
9.26.2.1	<code>TableMismatchedRecordLength</code>	80
9.27	<code>gldb::TableNoSuchField</code> Class Reference	80
9.27.1	Detailed Description	81
9.27.2	Constructor & Destructor Documentation	81
9.27.2.1	<code>TableNoSuchField</code>	81
9.28	<code>gldb::TableNoSuchRecord</code> Class Reference	81
9.28.1	Detailed Description	82
9.28.2	Constructor & Destructor Documentation	82
9.28.2.1	<code>TableNoSuchRecord</code>	82
9.29	<code>gldb::TableRow</code> Class Reference	82
9.29.1	Detailed Description	83
9.29.2	Constructor & Destructor Documentation	83

9.29.2.1	TableRow	83
9.29.2.2	TableRow	83
9.29.2.3	TableRow	83
9.29.2.4	TableRow	84
9.29.2.5	TableRow	84
9.29.2.6	TableRow	84
9.29.2.7	~TableRow	84
9.29.3	Member Function Documentation	84
9.29.3.1	append_field	84
9.29.3.2	append_field	84
9.29.3.3	append_field	84
9.29.3.4	append_field	85
9.29.3.5	append_field	85
9.29.3.6	operator=	85
9.29.3.7	operator=	85
9.29.3.8	operator[]	85
9.29.3.9	operator[]	86
9.29.3.10	print	86
9.29.3.11	record_string	86
9.29.3.12	record_string	86
9.29.3.13	size	86
9.29.4	Member Data Documentation	87
9.29.4.1	m_fields	87
10	File Documentation	89
10.1	lib/config/config.cpp File Reference	89
10.1.1	Detailed Description	89
10.2	lib/config/config.h File Reference	90
10.2.1	Detailed Description	91
10.3	lib/config/config_getopt.cpp File Reference	91
10.3.1	Detailed Description	91
10.3.2	Macro Definition Documentation	92
10.3.2.1	_XOPEN_SOURCE	92
10.4	lib/database/data_structures.h File Reference	92
10.4.1	Detailed Description	93
10.5	lib/database/database.h File Reference	93
10.5.1	Detailed Description	94
10.6	lib/database/dbconn.cpp File Reference	95
10.6.1	Detailed Description	95
10.7	lib/database/dbconn.h File Reference	96

10.7.1 Detailed Description	97
10.8 lib/database/dbconnimp.h File Reference	97
10.8.1 Detailed Description	99
10.9 lib/database/table.cpp File Reference	99
10.9.1 Detailed Description	99
10.10lib/database/table.h File Reference	100
10.10.1 Detailed Description	101
10.11lib/database/tablefield.cpp File Reference	101
10.11.1 Detailed Description	102
10.12lib/database/tablefield.h File Reference	102
10.12.1 Detailed Description	103
10.13lib/database/ablerow.cpp File Reference	103
10.13.1 Detailed Description	104
10.14lib/database/ablerow.h File Reference	104
10.14.1 Detailed Description	105
10.15lib/database_imp/database_imp.h File Reference	106
10.15.1 Detailed Description	107
10.16lib/database_imp/dummy/dbconn_dummy_imp.cpp File Reference	108
10.16.1 Detailed Description	108
10.17lib/database_imp/dummy/dbconn_dummy_imp.h File Reference	109
10.17.1 Detailed Description	110
10.18lib/database_imp/mysql/dbconn_mysql_functions.cpp File Reference	111
10.18.1 Detailed Description	111
10.19lib/database_imp/mysql/dbconn_mysql_imp.cpp File Reference	112
10.19.1 Detailed Description	112
10.20lib/database_imp/mysql/dbconn_mysql_imp.h File Reference	113
10.20.1 Detailed Description	114
10.21lib/database_imp/mysql/dbconn_mysql_result.cpp File Reference	115
10.21.1 Detailed Description	115
10.22lib/database_imp/mysql/dbconn_mysql_result.h File Reference	116
10.22.1 Detailed Description	116
10.23lib/dbsql/dbsql.h File Reference	117
10.23.1 Detailed Description	118
10.24lib/dbsql/dbsql_dummy.h File Reference	118
10.24.1 Detailed Description	119
10.25lib/dbsql/dbsql_implementations.h File Reference	120
10.25.1 Detailed Description	120
10.26lib/dbsql/dbsql_mysql.h File Reference	121
10.26.1 Detailed Description	122
10.27lib/dbsql/dbsqlstatements.cpp File Reference	122

10.27.1 Detailed Description	122
10.28lib/dbsql/dbsqlstatements.h File Reference	123
10.28.1 Detailed Description	124
10.29lib/gldb/gldatabase.cpp File Reference	124
10.29.1 Detailed Description	125
10.29.2 Function Documentation	125
10.29.2.1 boolstring_to_bool	125
10.30lib/gldb/gldatabase.h File Reference	125
10.30.1 Detailed Description	127
10.31lib/gldb/gldb.h File Reference	127
10.31.1 Detailed Description	128
10.32lib/gldb/glexception.h File Reference	128
10.32.1 Detailed Description	129
10.33lib/gldb/gluser.cpp File Reference	130
10.33.1 Detailed Description	130
10.34lib/gldb/gluser.h File Reference	130
10.34.1 Detailed Description	131
10.35lib/gldb/gluser_pass.cpp File Reference	131
10.35.1 Detailed Description	132
10.35.2 Macro Definition Documentation	132
10.35.2.1 _XOPEN_SOURCE	132
10.35.3 Function Documentation	133
10.35.3.1 generate_salt	133
10.36lib/stringhelp/stringhelp.cpp File Reference	133
10.36.1 Detailed Description	133
10.37lib/stringhelp/stringhelp.h File Reference	133
10.37.1 Detailed Description	135
10.38progs/gl_db/gl_db_main.cpp File Reference	135
10.38.1 Detailed Description	136
10.39progs/gl_report/gl_report_main.cpp File Reference	136
10.39.1 Detailed Description	138
10.40progs/gl_user/gl_user_main.cpp File Reference	138
10.40.1 Detailed Description	139
10.40.2 Function Documentation	139
10.40.2.1 check_db_parameters	139
10.40.2.2 check_help_and_version	140
10.40.2.3 check_user_password	140
10.40.2.4 enable_user	140
10.40.2.5 get_user	140
10.40.2.6 login	140

10.40.2.7 main	141
10.40.2.8 set_configuration	141
10.40.2.9 set_user_password	141
10.40.2.10 show_user_details	141

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

Bug List

Member `gldb::Table::Table` (`const Table &table`)

'explicit' removed from here after failure to compile at end of MySQL query function.

Chapter 4

Module Index

4.1 Modules

Here is a list of all modules:

General Ledger database module.	15
Database interaction module	16
SQL statements module	19
Program configuration module	20
General purpose helpers.	21
Reporting program.	23
Database program.	25

Chapter 5

Class Index

5.1 Class Hierarchy

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

genleg::Config	27
genleg::ConfigException	33
genleg::ConfigBadConfigFile	29
genleg::ConfigBadOption	31
genleg::ConfigCouldNotOpenFile	32
genleg::ConfigOptionNotSet	34
gldb::DBConn	35
gldb::DBConnException	42
gldb::DBConnCouldNotConnect	37
gldb::DBConnCouldNotQuery	38
gldb::DBConnImp	43
gldb::DBConnDummy	40
gldb::DBConnMySQL	44
genleg::DBSQLStatements	49
genleg::DBSQLDummy	47
genleg::DBSQLMySQL	48
genleg::GLDatabase	52
genleg::GLDBException	57
genleg::GLUser	58
gldb::MySQLResult	63
gldb::Table	64
gldb::TableException	72
gldb::TableBadInputFile	70
gldb::TableCouldNotOpenInputFile	71
gldb::TableMismatchedRecordLength	79
gldb::TableNoSuchField	80
gldb::TableNoSuchRecord	81
gldb::TableField	73
gldb::TableRow	82

Chapter 6

Class Index

6.1 Class List

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

genleg::Config	Configuration options class	27
genleg::ConfigBadConfigFile	Exception class for badly formed configuration file	29
genleg::ConfigBadOption	Exception class for bad provided option	31
genleg::ConfigCouldNotOpenFile	Exception class for when conf file cannot be opened	32
genleg::ConfigException	Configuration module exception base class	33
genleg::ConfigOptionNotSet	Exception class for option not set	34
gldb::DBConn	Database connection class	35
gldb::DBConnCouldNotConnect	Could not connect to database exception class	37
gldb::DBConnCouldNotQuery	Could not execute database query exception class	38
gldb::DBConnDummy	Dummy database implementation class	40
gldb::DBConnException	Base database connection exception class	42
gldb::DBConnImp	Abstract database implementation base class	43
gldb::DBConnMySQL	MySQL database implementation class	44
genleg::DBSQLDummy	Dummy SQL statements class	47
genleg::DBSQLMySQL	MySQL SQL statements class	48
genleg::DBSQLStatements	SQL statements class	49
genleg::GLDatabase	General ledger database class	52
genleg::GLDBException	Base general ledger database exceptionc class	57
genleg::GLUser	General ledger user class	58

gldb::MySQLResult	
MySQL result structure class	63
gldb::Table	
Database table class	64
gldb::TableBadInputFile	
Could not connect to database exception class	70
gldb::TableCouldNotOpenInputFile	
Could not connect to database exception class	71
gldb::TableException	
Base database connection exception class	72
gldb::TableField	
Database table field class	73
gldb::TableMismatchedRecordLength	
Mismatched record length exception class	79
gldb::TableNoSuchField	
No such field exception class	80
gldb::TableNoSuchRecord	
No such record exception class	81
gldb::TableRow	
Database table row class	82

Chapter 7

File Index

7.1 File List

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

lib/config/ config.cpp	
Implementation of program configurations class	89
lib/config/ config.h	
Interface to program configurations class	90
lib/config/ config_getopt.cpp	
Implementation of command line functionality	91
lib/database/ data_structures.h	
Main interface to database data structures	92
lib/database/ database.h	
User interface to database functionality	93
lib/database/ dbconn.cpp	
Implementation of database connection class	95
lib/database/ dbconn.h	
Interface to database connection base class	96
lib/database/ dbconnimp.h	
Interface to abstract database implementation base class	97
lib/database/ table.cpp	
Implementation of database table data structure	99
lib/database/ table.h	
Interface to database table data structure	100
lib/database/ tablefield.cpp	
Implementation of database table field class	101
lib/database/ tablefield.h	
Interface to database table field class	102
lib/database/ tablerow.cpp	
Implementation of database table row data structure	103
lib/database/ tablerow.h	
Interface to database table row data structure	104
lib/database_imp/ database_imp.h	
Interface to database implementation factory function	106
lib/database_imp/dummy/ dbconn_dummy_imp.cpp	
Implementation of Dummy database connection implementation class	108
lib/database_imp/dummy/ dbconn_dummy_imp.h	
Interface to dummy database connection implementation class	109
lib/database_imp/mysql/ dbconn_mysql_functions.cpp	
Implementation of MySQL implementation factory function	111
lib/database_imp/mysql/ dbconn_mysql_imp.cpp	
Implementation of MySQL database connection implementation class	112

lib/database_imp/mysql/dbconn_mysql_imp.h	
Interface to MySQL database connection implementation class	113
lib/database_imp/mysql/dbconn_mysql_result.cpp	
Implementation of MySQL result structure resource handle class	115
lib/database_imp/mysql/dbconn_mysql_result.h	
Interface to MySQL result structure resource handle class	116
lib/dbsql/dbsql.h	
User interface to DBSQL module	117
lib/dbsql/dbsql_dummy.h	
Interface to dummy SQL statement class	118
lib/dbsql/dbsql_functions.h	??
lib/dbsql/dbsql_implementations.h	
Aggregation header for DBSqlStatements implementations	120
lib/dbsql/dbsql_mysql.h	
Interface to MySQL SQL statement class	121
lib/dbsql/dbsqlstatements.cpp	
Implementation of SQL statement class	122
lib/dbsql/dbsqlstatements.h	
Implementation of SQL module standalone functions	123
lib/gldb/gldatabase.cpp	
Implementation of General Ledger database class	124
lib/gldb/gldatabase.h	
Interface to General Ledger database class	125
lib/gldb/gldb.h	
User interface to General Ledger database module	127
lib/gldb/glexception.h	
Interface to General Ledger base exception class	128
lib/gldb/gluser.cpp	
Implementation of user class	130
lib/gldb/gluser.h	
Interface to user class	130
lib/gldb/gluser_pass.cpp	
Implementation of password functions for user class	131
lib/stringhelp/stringhelp.cpp	
Implementation of string helper functions	133
lib/stringhelp/stringhelp.h	
Interface to string helper functions	133
progs/gl_db/gl_db_main.cpp	
Main functionality for gl_db program	135
progs/gl_report/gl_report_main.cpp	
Main functionality for gl_report program	136
progs/gl_user/gl_user_main.cpp	
Main functionality for gl_user program	138

Chapter 8

Module Documentation

8.1 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.

8.1.1 Detailed Description

Module for interacting with the general ledger database model.

8.2 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.
- class [gldb::MySQLResult](#)
MySQL result structure 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.
- static [TableRow get_field_names](#) (MySQLResult &result)
Gets field names from a MySQL result structure.
- static [TableRow get_row](#) (MySQLResult &result, MYSQL_ROW row)
Creates a TableRow from a MySQL result row.

8.2.1 Detailed Description

Module for interacting with the database.

8.2.2 Function Documentation

8.2.2.1 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.

8.2.2.2 std::string glldb::get_database_type ()

Returns the name of the compiled-in database type.

Returns

The name of the compiled-in database type.

8.2.2.3 static TableRow get_field_names (MySQLResult & *result*) [static]

Gets field names from a MySQL result structure.

Parameters

<i>result</i>	The MySQL result structure.
---------------	-----------------------------

Returns

A TableRow containing the field names.

8.2.2.4 static TableRow get_row (MySQLResult & *result*, MYSQL_ROW *row*) [static]

Creates a TableRow from a MySQL result row.

Parameters

<i>result</i>	The MySQL result structure.
<i>row</i>	The MySQL row structure.

Returns

A TableRow containing the row data.

8.3 SQL statements module

Classes

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

8.3.1 Detailed Description

Module for producing SQL statements used by program.

8.4 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.

8.4.1 Detailed Description

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

8.5 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.

8.5.1 Detailed Description

General purpose helper classes and functions.

8.5.2 Function Documentation

8.5.2.1 `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.

8.5.2.2 `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`.

8.5.2.3 `std::string & pgstring::trim (std::string & s)`

Trims leading and trailing whitespace from a string.

Parameters

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

Returns

The trimmed string.

8.5.2.4 `std::string & pgstring::trim_back (std::string & s)`

Trims trailing whitespace from a string.

Parameters

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

Returns

The trimmed string.

8.5.2.5 `std::string & pgstring::trim_front (std::string & s)`

Trims leading whitespace from a string.

Parameters

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

Returns

The trimmed string.

8.6 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.

8.6.1 Detailed Description

Administrative reporting program.

8.6.2 Function Documentation

8.6.2.1 static std::string login (void) [static]

Gets a password from the terminal.

Returns

The password.

8.6.2.2 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.

8.6.2.3 `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> .

8.7 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.

8.7.1 Detailed Description

Administrative database management program.

8.7.2 Function Documentation

8.7.2.1 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.

8.7.2.2 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.

8.7.2.3 static std::string login (void) [static]

Gets a password from the terminal.

Returns

The password.

8.7.2.4 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.

8.7.2.5 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 9

Class Documentation

9.1 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](#)

9.1.1 Detailed Description

Configuration options class.

9.1.2 Constructor & Destructor Documentation

9.1.2.1 Config::Config ()

Constructor

9.1.2.2 Config::~Config ()

Destructor

9.1.3 Member Function Documentation

9.1.3.1 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.

9.1.3.2 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.

9.1.3.3 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

<i>ConfigOptionNotSet</i>	If the named option has not been set.
---	---------------------------------------

9.1.3.4 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

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

9.1.3.5 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

<i>ConfigCouldNotOpenFile</i>	If the configuration file cannot be opened.
<i>ConfigBadConfigFile</i>	If the configuration file is badly formed.

9.1.4 Member Data Documentation

9.1.4.1 std::map<std::string, std::string> genleg::Config::m_opts_set [private]

Map of options which have been set

9.1.4.2 std::list<std::pair<std::string, enum Argument> > genleg::Config::m_opts_supp [private]

List of options which are supported

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

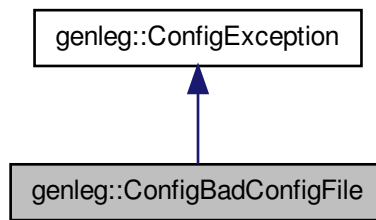
- lib/config/[config.h](#)
- lib/config/[config.cpp](#)
- lib/config/[config_getopt.cpp](#)

9.2 genleg::ConfigBadConfigFile Class Reference

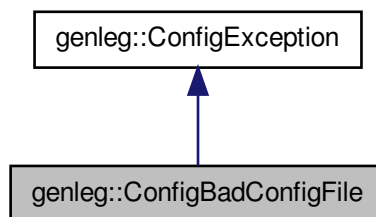
Exception class for badly formed configuration file.

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

Inheritance diagram for `genleg::ConfigBadConfigFile`:



Collaboration diagram for `genleg::ConfigBadConfigFile`:



Public Member Functions

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

9.2.1 Detailed Description

Exception class for badly formed configuration file.

9.2.2 Constructor & Destructor Documentation

9.2.2.1 `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:

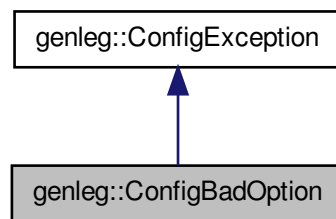
- [lib/config/config.h](#)

9.3 genleg::ConfigBadOption Class Reference

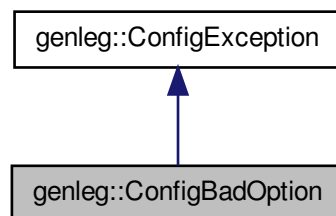
Exception class for bad provided option.

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

Inheritance diagram for genleg::ConfigBadOption:



Collaboration diagram for genleg::ConfigBadOption:



Public Member Functions

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

9.3.1 Detailed Description

Exception class for bad provided option.

9.3.2 Constructor & Destructor Documentation

9.3.2.1 `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:

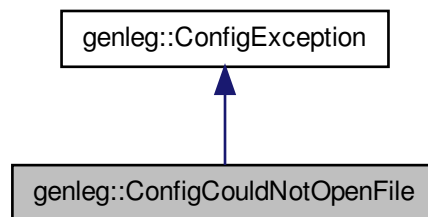
- `lib/config/config.h`

9.4 `genleg::ConfigCouldNotOpenFile` Class Reference

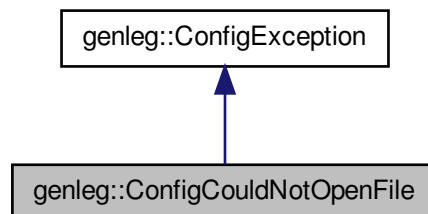
Exception class for when conf file cannot be opened.

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

Inheritance diagram for `genleg::ConfigCouldNotOpenFile`:



Collaboration diagram for `genleg::ConfigCouldNotOpenFile`:



Public Member Functions

- `ConfigCouldNotOpenFile (const std::string &msg)`
Constructor.

9.4.1 Detailed Description

Exception class for when conf file cannot be opened.

9.4.2 Constructor & Destructor Documentation

9.4.2.1 `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:

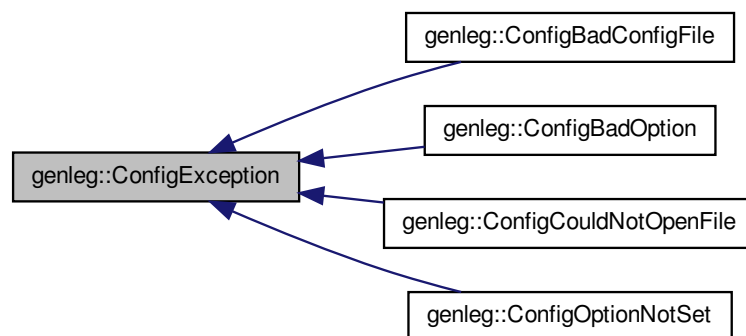
- [lib/config/config.h](#)

9.5 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.

9.5.1 Detailed Description

Configuration module exception base class.

9.5.2 Constructor & Destructor Documentation

9.5.2.1 `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:

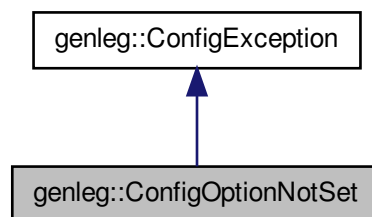
- `lib/config/config.h`

9.6 `genleg::ConfigOptionNotSet` Class Reference

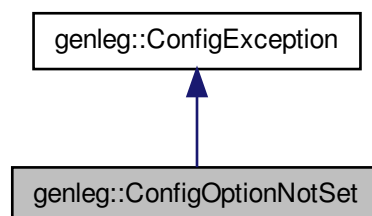
Exception class for option not set.

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

Inheritance diagram for `genleg::ConfigOptionNotSet`:



Collaboration diagram for `genleg::ConfigOptionNotSet`:



Public Member Functions

- `ConfigOptionNotSet (const std::string &msg)`

Constructor.

9.6.1 Detailed Description

Exception class for option not set.

9.6.2 Constructor & Destructor Documentation

9.6.2.1 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:

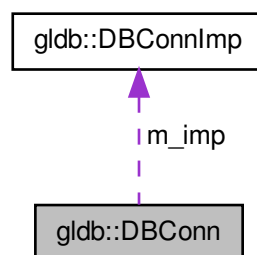
- lib/config/[config.h](#)

9.7 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](#) (const std::string &sql_query)
Runs an SQL query.
- [Table select](#) (const std::string &query)

Runs an SQL SELECT query.

- [DBConn](#) (const [DBConn](#) &)
- [DBConn](#) (const [DBConn](#) &&)
- [DBConn](#) & [operator=](#) (const [DBConn](#) &)
- [DBConn](#) & [operator=](#) (const [DBConn](#) &&)

Private Attributes

- [DBConnImp](#) * [m_imp](#)

9.7.1 Detailed Description

Database connection class.

9.7.2 Constructor & Destructor Documentation

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

Constructor.

Parameters

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

9.7.2.2 [gldb::DBConn::DBConn \(const \[DBConn\]\(#\) & \)](#)

Deleted copy constructor

9.7.2.3 [gldb::DBConn::DBConn \(const \[DBConn\]\(#\) && \)](#)

Deleted move constructor

9.7.3 Member Function Documentation

9.7.3.1 [DBConn& gldb::DBConn::operator= \(const \[DBConn\]\(#\) & \)](#)

Deleted copy assignment operator

9.7.3.2 [DBConn& gldb::DBConn::operator= \(const \[DBConn\]\(#\) && \)](#)

Deleted move assignment operator

9.7.3.3 [void DBConn::query \(const std::string & *sql_query* \)](#)

Runs an SQL query.

Parameters

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

Returns

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

9.7.3.4 Table DBConn::select (const std::string & *query*)

Runs an SQL SELECT query.

Parameters

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

Returns

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

9.7.4 Member Data Documentation

9.7.4.1 DBConnImp* glldb::DBConn::m_imp [private]

Pointer to database implementation object.

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

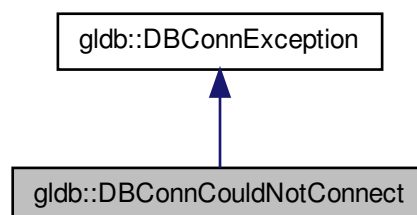
- lib/database/[dbconn.h](#)
- lib/database/[dbconn.cpp](#)

9.8 glldb::DBConnCouldNotConnect Class Reference

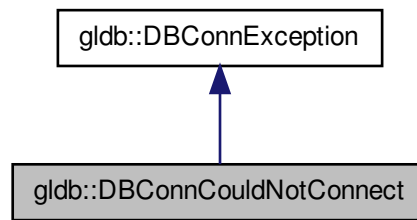
Could not connect to database exception class.

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

Inheritance diagram for glldb::DBConnCouldNotConnect:



Collaboration diagram for `gldb::DBConnCouldNotConnect`:



Public Member Functions

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

9.8.1 Detailed Description

Could not connect to database exception class.

9.8.2 Constructor & Destructor Documentation

9.8.2.1 `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:

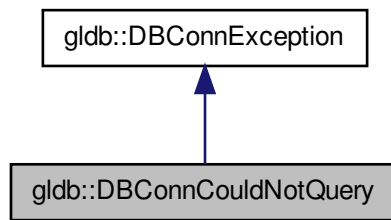
- lib/database/[dbconn.h](#)

9.9 gldb::DBConnCouldNotQuery Class Reference

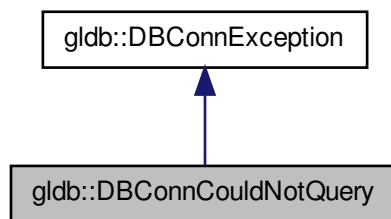
Could not execute database query exception class.

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

Inheritance diagram for glldb::DBConnCouldNotQuery:



Collaboration diagram for glldb::DBConnCouldNotQuery:



Public Member Functions

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

9.9.1 Detailed Description

Could not execute database query exception class.

9.9.2 Constructor & Destructor Documentation

9.9.2.1 `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:

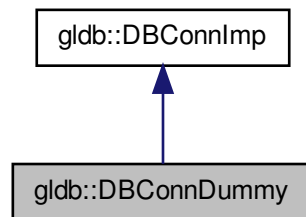
- [lib/database/dbconn.h](#)

9.10 gldb::DBConnDummy Class Reference

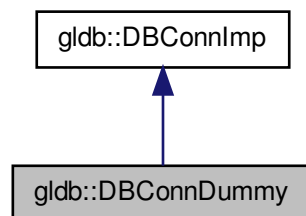
Dummy database implementation class.

```
#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](#) &)
- virtual void [query](#) (const std::string &sql_query)
Runs an SQL query.
- [Table select](#) (const std::string &[query](#))
Fakes running of an SQL SELECT query.

9.10.1 Detailed Description

Dummy database implementation class.

9.10.2 Constructor & Destructor Documentation

9.10.2.1 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.

9.10.2.2 glldb::DBConnDummy::DBConnDummy (const DBConnDummy &)

Deleted copy constructor

9.10.2.3 DBConnDummy::~DBConnDummy () [virtual]

Destructor

9.10.3 Member Function Documentation

9.10.3.1 DBConnDummy& glldb::DBConnDummy::operator= (const DBConnDummy &)

Deleted assignment operator

9.10.3.2 void DBConnDummy::query (const std::string & *sql_query*) [virtual]

Runs an SQL query.

Parameters

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

Exceptions

<i>DBConnCouldNotQuery</i>	If could not successfully execute query.
--	--

Implements [`glldb::DBConnImp`](#).

9.10.3.3 Table DBConnDummy::select (const 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:

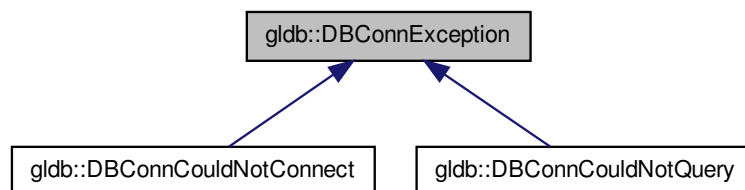
- [lib/database_imp/dummy/dbconn_dummy_imp.h](#)
- [lib/database_imp/dummy/dbconn_dummy_imp.cpp](#)

9.11 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.

9.11.1 Detailed Description

Base database connection exception class.

9.11.2 Constructor & Destructor Documentation

9.11.2.1 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:

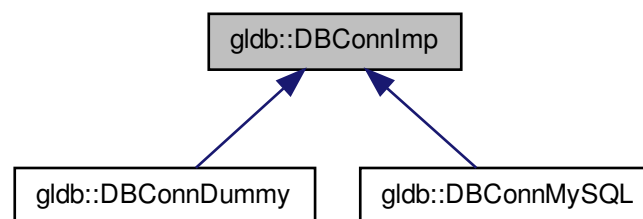
- lib/database/[dbconn.h](#)

9.12 glldb::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](#) (const std::string &sql_query)=0
Runs an SQL query.
- virtual [Table select](#) (const std::string &query)=0
Runs an SQL SELECT query.

9.12.1 Detailed Description

Abstract database implementation base class.

9.12.2 Constructor & Destructor Documentation

9.12.2.1 glldb::DBConnImp::DBConnImp () [inline]

Constructor

9.12.2.2 virtual glldb::DBConnImp::~~DBConnImp () [inline], [virtual]

Destructor

9.12.3 Member Function Documentation

9.12.3.1 `virtual void gldb::DBConnImp::query (const std::string & sql_query) [pure virtual]`

Runs an SQL query.

Parameters

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

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

9.12.3.2 `virtual Table gldb::DBConnImp::select (const 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 [gldb::DBConnMySQL](#), and [gldb::DBConnDummy](#).

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

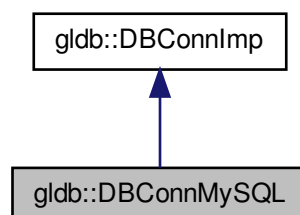
- `lib/database/dbconnimp.h`

9.13 gldb::DBConnMySQL Class Reference

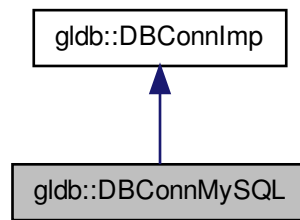
MySQL database implementation class.

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

Inheritance diagram for `gldb::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](#) &)
- [DBConnMySQL](#) (const [DBConnMySQL](#) &&)
- virtual [~DBConnMySQL](#) ()
- [DBConnMySQL](#) & [operator=](#) (const [DBConnMySQL](#) &)
- [DBConnMySQL](#) & [operator=](#) (const [DBConnMySQL](#) &&)
- virtual void [query](#) (const std::string &sql_query)
Runs an SQL query.
- virtual [Table select](#) (const std::string &sql_query)
Runs an SQL SELECT query.

Private Attributes

- MYSQL * [m_conn](#)

9.13.1 Detailed Description

MySQL database implementation class.

9.13.2 Constructor & Destructor Documentation

9.13.2.1 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

<i>DBConnCouldNotConnect</i>	If could not connect to database.
--	-----------------------------------

9.13.2.2 `gldb::DBConnMySQL::DBConnMySQL (const DBConnMySQL &)`

Deleted copy constructor

9.13.2.3 `gldb::DBConnMySQL::DBConnMySQL (const DBConnMySQL &&)`

Delete move constructor

9.13.2.4 `virtual gldb::DBConnMySQL::~DBConnMySQL () [virtual]`

Destructor

9.13.3 Member Function Documentation

9.13.3.1 `DBConnMySQL& gldb::DBConnMySQL::operator= (const DBConnMySQL &)`

Deleted assignment operator

9.13.3.2 `DBConnMySQL& gldb::DBConnMySQL::operator= (const DBConnMySQL &&)`

Deleted move assignment operator

9.13.3.3 `virtual void gldb::DBConnMySQL::query (const std::string & sql_query) [virtual]`

Runs an SQL query.

Parameters

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

Exceptions

<i>DBConnCouldNotQuery</i>	If could not successfully execute query.
--	--

Implements [`gldb::DBConnImp`](#).

9.13.3.4 `virtual Table gldb::DBConnMySQL::select (const std::string & sql_query) [virtual]`

Runs an SQL SELECT query.

Parameters

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

Returns

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

Exceptions

<i>DBConnCouldNotQuery</i>	If could not successfully execute query.
--	--

Implements [gldb::DBConnImp](#).

9.13.4 Member Data Documentation

9.13.4.1 MySQL* gldb::DBConnMySQL::m_conn [private]

The initialized MySQL handle.

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

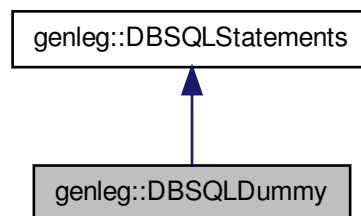
- lib/database_imp/mysql/[dbconn_mysql_imp.h](#)
- lib/database_imp/mysql/[dbconn_mysql_imp.cpp](#)

9.14 genleg::DBSQLDummy Class Reference

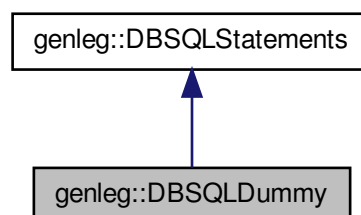
Dummy SQL statements class.

```
#include <dbsql_dummy.h>
```

Inheritance diagram for genleg::DBSQLDummy:



Collaboration diagram for genleg::DBSQLDummy:



Additional Inherited Members

9.14.1 Detailed Description

Dummy SQL statements class.

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

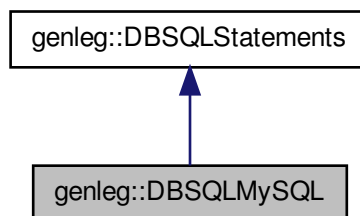
- `lib/dbsql/dbsql_dummy.h`

9.15 genleg::DBSQLMySQL Class Reference

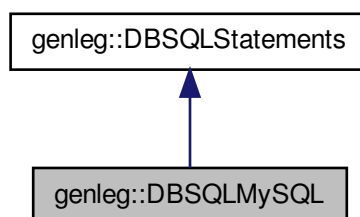
MySQL SQL statements class.

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

Inheritance diagram for genleg::DBSQLMySQL:



Collaboration diagram for genleg::DBSQLMySQL:



Additional Inherited Members

9.15.1 Detailed Description

MySQL SQL statements class.

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

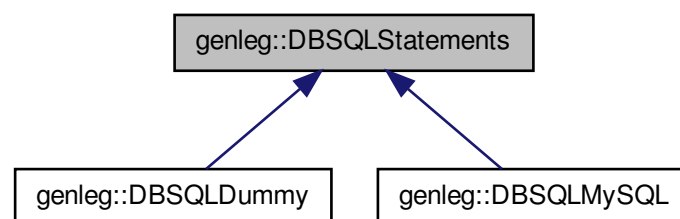
- lib/dbsql/dbsql_mysql.h

9.16 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.

9.16.1 Detailed Description

SQL statements class.

9.16.2 Constructor & Destructor Documentation

9.16.2.1 DBSQLStatements::DBSQLStatements ()

Constructor

9.16.2.2 DBSQLStatements::~~DBSQLStatements () [virtual]

Destructor

9.16.3 Member Function Documentation

9.16.3.1 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.

9.16.3.2 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.

9.16.3.3 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.

9.16.3.4 `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.

9.16.3.5 `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.

9.16.3.6 `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.

9.16.3.7 `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.

9.16.3.8 `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.

9.16.3.9 `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.

9.16.3.10 `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:

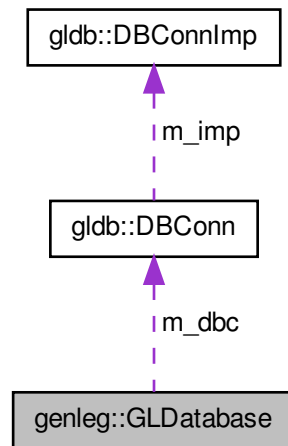
- [lib/dbsql/dbsqlstatements.h](#)
- [lib/dbsql/dbsqlstatements.cpp](#)

9.17 genleg::GLDatabase Class Reference

General ledger database class.

```
#include <glatabase.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`

9.17.1 Detailed Description

General ledger database class.

9.17.2 Constructor & Destructor Documentation

9.17.2.1 `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

GLDBException	on error.
-------------------------------	-----------

9.17.2.2 `GLDatabase::~~GLDatabase ()`

Destructor

9.17.3 Member Function Documentation

9.17.3.1 `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.

9.17.3.2 void GLDatabase::create_structure ()

Creates the database structure.

Exceptions

GLDBException	on error.
-------------------------------	-----------

9.17.3.3 GLUser GLDatabase::create_user (gldb::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.

9.17.3.4 void GLDatabase::destroy_structure ()

Destroys the database structure.

Exceptions

GLDBException	on error.
-------------------------------	-----------

9.17.3.5 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

GLDBException	if the user cannot be found.
-------------------------------	------------------------------

9.17.3.6 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

<i>GLDBException</i>	if the user cannot be found.
--------------------------------------	------------------------------

9.17.3.7 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.

9.17.3.8 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

<i>GLDBException</i>	on error.
--------------------------------------	-----------

9.17.3.9 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.

9.17.3.10 void GLDatabase::update_user (const GLUser & *user*)

Updates a user's details.

Parameters

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

9.17.4 Member Data Documentation

9.17.4.1 `gldb::DBConn genleg::GLDatabase::m_dbc` `[private]`

Database connection

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

SQL statements object

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

Vector containing database table names

9.17.4.4 `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](#)

9.18 genleg::GLDBException Class Reference

Base general ledger database exceptionc class.

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

Public Member Functions

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

9.18.1 Detailed Description

Base general ledger database exceptionc class.

9.18.2 Constructor & Destructor Documentation

9.18.2.1 `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:

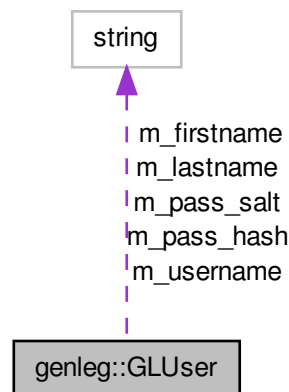
- [lib/gldb/glexception.h](#)

9.19 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](#)

9.19.1 Detailed Description

General ledger user class.

9.19.2 Constructor & Destructor Documentation

- 9.19.2.1 **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.

- 9.19.2.2 **GLUser::~~GLUser ()**

Destructor

9.19.3 Member Function Documentation

9.19.3.1 `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.

9.19.3.2 `bool GLUser::enabled () const`

Returns the user's enabled status.

Returns

The user's enabled status.

9.19.3.3 `const std::string & GLUser::firstname () const`

Returns the user's first name.

Returns

The user's first name.

9.19.3.4 `const std::string & GLUser::id () const`

Returns the user ID.

Returns

The user ID.

9.19.3.5 `const std::string & GLUser::lastname () const`

Returns the user's last name.

Returns

The user's last name.

9.19.3.6 `const std::string & GLUser::pass_hash () const`

Returns the user's hashed password.

Returns

The user's hashed password.

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

Returns the user's password salt.

Returns

The user's password salt.

9.19.3.8 `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.

9.19.3.9 `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.
--------------------	--------------------------------

9.19.3.10 `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.
----------------------	----------------------------

9.19.3.11 `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.
---------------------	---------------------------

9.19.3.12 `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.
-----------------	---

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

Sets a user's username.

Parameters

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

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

Returns the username.

Returns

The username.

9.19.4 Member Data Documentation

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

User's enabled status

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

User's first name

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

User ID

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

User's last name

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

User's hashed password

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

User's password salt

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

List of permissions

9.19.4.8 `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](#)

9.20 gldb::MySQLResult Class Reference

MySQL result structure class.

```
#include <dbconn_mysql_result.h>
```

Public Member Functions

- [MySQLResult](#) (MySQL *conn)
Constructor.
- [~MySQLResult](#) ()
- [MySQLResult](#) (const [MySQLResult](#) &result)
- [MySQLResult](#) ([MySQLResult](#) &&result)
- [MySQLResult](#) & operator= (const [MySQLResult](#) &result)
- [MySQLResult](#) & operator= ([MySQLResult](#) &&result)
- MYSQL_RES * [result](#) ()
Returns the MYSQL_RES pointer.
- unsigned int [num_fields](#) () const
Returns the number of fields in the result set.

Private Attributes

- MYSQL_RES * [m_result](#)
- unsigned int [m_num_fields](#)

9.20.1 Detailed Description

MySQL result structure class.

9.20.2 Constructor & Destructor Documentation

9.20.2.1 MySQLResult::MySQLResult (MySQL * conn) [explicit]

Constructor.

Parameters

<i>conn</i>	MySQL connection
-------------	------------------

Exceptions

DBConnCouldNotQuery	on failure
-------------------------------------	------------

9.20.2.2 gldb::MySQLResult::~~MySQLResult ()

Destructor

9.20.2.3 gldb::MySQLResult::MySQLResult (const MySQLResult & result)

Deleted copy constructor

9.20.2.4 `gldb::MySQLResult::MySQLResult (MySQLResult && result)`

Deleted move constructor

9.20.3 Member Function Documentation

9.20.3.1 `unsigned int glldb::MySQLResult::num_fields () const [inline]`

Returns the number of fields in the result set.

Returns

The number of fields in the result set.

9.20.3.2 `MySQLResult& glldb::MySQLResult::operator= (const MySQLResult & result)`

Deleted copy assignment operator

9.20.3.3 `MySQLResult& glldb::MySQLResult::operator= (MySQLResult && result)`

Deleted move assignment operator

9.20.3.4 `MYSQL_RES* glldb::MySQLResult::result () [inline]`

Returns the MYSQL_RES pointer.

Returns

The MYSQL_RES pointer.

9.20.4 Member Data Documentation

9.20.4.1 `unsigned int glldb::MySQLResult::m_num_fields [private]`

The number of fields in the result set

9.20.4.2 `MYSQL_RES* glldb::MySQLResult::m_result [private]`

The MYSQL_RES pointer

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

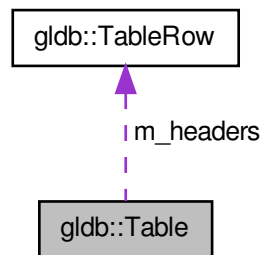
- [lib/database_imp/mysql/dbconn_mysql_result.h](#)
- [lib/database_imp/mysql/dbconn_mysql_result.cpp](#)

9.21 `gldb::Table` Class Reference

Database table class.

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

Collaboration diagram for glldb::Table:



Public Member Functions

- [Table](#) (const [TableRow](#) &headers)
Constructor.
- [Table](#) ([TableRow](#) &&headers)
Constructor with move semantics.
- [Table](#) (const [Table](#) &table)
Copy constructor.
- [Table](#) ([Table](#) &&table)
Move constructor.
- [Table](#) & [operator=](#) (const [Table](#) &table)
Copy assignment operator.
- [Table](#) & [operator=](#) ([Table](#) &&table)
Move assignment operator.
- [~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](#) (const std::vector< bool > &vec)
Sets the quote flags for the records.
- void [set_quoted](#) (std::vector< bool > &&vec)
Sets the quote flags for the records with move semantics.
- 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.
- void [append_record](#) ([TableRow](#) &&new_record)
Appends a record to the table with move semantics.
- 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](#)

9.21.1 Detailed Description

Database table class.

9.21.2 Constructor & Destructor Documentation

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

Constructor.

Parameters

<i>headers</i>	Table row containing field names.
----------------	---

9.21.2.2 `Table::Table (TableRow && headers) [explicit]`

Constructor with move semantics.

Parameters

<i>headers</i>	Table row containing field names.
----------------	---

9.21.2.3 `Table::Table (const Table & table)`

Copy constructor.

Bug 'explicit' removed from here after failure to compile at end of MySQL query function.

Parameters

<i>table</i>	Table to copy.
--------------	--------------------------------

9.21.2.4 `Table::Table (Table && table) [explicit]`

Move constructor.

Parameters

<i>table</i>	Table to move.
--------------	--------------------------------

9.21.2.5 Table::~~Table ()

Destructor

9.21.3 Member Function Documentation

9.21.3.1 void Table::append_record (const TableRow & new_record)

Appends a record to the table.

Parameters

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

9.21.3.2 void Table::append_record (TableRow && new_record)

Appends a record to the table with move semantics.

Parameters

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

9.21.3.3 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

<i>TableBadInputFile</i>	on badly formed input file.
<i>TableCouldNotOpenInputFile</i>	on bad filename.

9.21.3.4 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

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

9.21.3.5 `const TableRow & Table::get_headers () const`

Returns the field names.

Returns

The field names.

9.21.3.6 `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.

9.21.3.7 `size_t Table::num_fields () const`

Returns the number of fields in each row.

Returns

The number of fields in each row.

9.21.3.8 `size_t Table::num_records () const`

Returns the number of record in the table.

Returns

The number of records in the table.

9.21.3.9 `Table & Table::operator= (const Table & table)`

Copy assignment operator.

Parameters

<i>table</i>	Table to copy.
--------------	--------------------------------

Returns

Reference to the assigned-to table.

9.21.3.10 Table & Table::operator= (Table && *table*)

Move assignment operator.

Parameters

<i>table</i>	Table to move.
--------------	----------------

Returns

Reference to the assigned-to table.

9.21.3.11 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.

9.21.3.12 void Table::set_quoted (const 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.
------------	---

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

Sets the quote flags for the records with move semantics.

Parameters

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

9.21.4 Member Data Documentation**9.21.4.1 TableRow glldb::Table::m_headers [private]**

The names of the fields

9.21.4.2 `std::vector<bool> gldb::Table::m_quoted` [private]

A vector to show if fields should be quoted for INSERT

9.21.4.3 `std::vector<TableRow> gldb::Table::m_records` [private]

A vector of the records

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

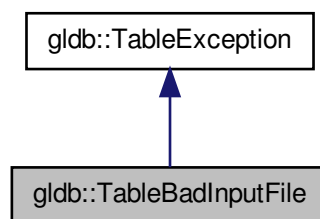
- [lib/database/table.h](#)
- [lib/database/table.cpp](#)

9.22 `gldb::TableBadInputFile` Class Reference

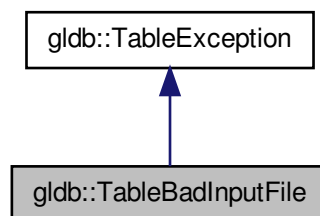
Could not connect to database exception class.

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

Inheritance diagram for `gldb::TableBadInputFile`:



Collaboration diagram for `gldb::TableBadInputFile`:



Public Member Functions

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

Constructor.

9.22.1 Detailed Description

Could not connect to database exception class.

9.22.2 Constructor & Destructor Documentation

9.22.2.1 glldb::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:

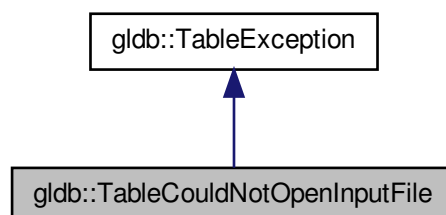
- lib/database/[table.h](#)

9.23 glldb::TableCouldNotOpenInputFile Class Reference

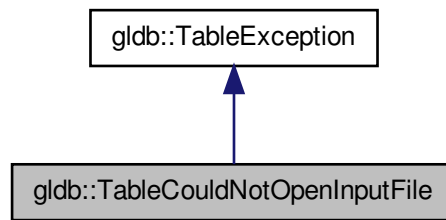
Could not connect to database exception class.

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

Inheritance diagram for glldb::TableCouldNotOpenInputFile:



Collaboration diagram for `gldb::TableCouldNotOpenInputFile`:



Public Member Functions

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

9.23.1 Detailed Description

Could not connect to database exception class.

9.23.2 Constructor & Destructor Documentation

9.23.2.1 `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:

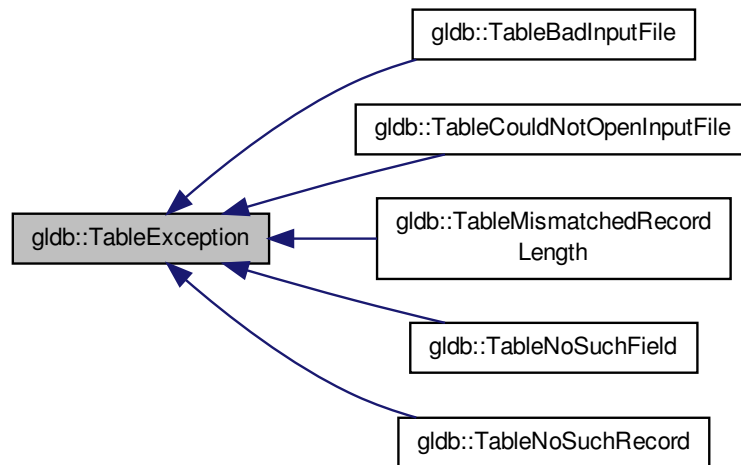
- `lib/database/table.h`

9.24 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.

9.24.1 Detailed Description

Base database connection exception class.

9.24.2 Constructor & Destructor Documentation

9.24.2.1 `gldb::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:

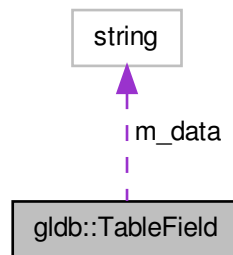
- lib/database/[table.h](#)

9.25 gldb::TableField Class Reference

Database table field class.

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

Collaboration diagram for `gldb::TableField`:



Public Member Functions

- [TableField](#) (const char *data)
*Constructor accepting `const char *` data.*
- [TableField](#) (const std::string &data)
Constructor accepting `std::string` data.
- [TableField](#) (std::string &&data)
Constructor accepting `std::string` data with move semantics.
- [TableField](#) (const [TableField](#) &field)
Copy constructor.
- [TableField](#) ([TableField](#) &&field)
Move constructor.
- [~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`.
- [TableField](#) & [operator=](#) (std::string &&data)
Overridden assignment operator for `std::string` with move semantics.
- [TableField](#) & [operator=](#) (const [TableField](#) &field)
Overridden copy assignment operator.
- [TableField](#) & [operator=](#) ([TableField](#) &&field)
Overridden move assignment operator.
- 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.

9.25.1 Detailed Description

Database table field class.

9.25.2 Constructor & Destructor Documentation

9.25.2.1 TableField::TableField (const char * data) [explicit]

Constructor accepting `const char * data`.

Parameters

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

9.25.2.2 TableField::TableField (const std::string & data) [explicit]

Constructor accepting `std::string data`.

Parameters

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

9.25.2.3 TableField::TableField (std::string && data) [explicit]

Constructor accepting `std::string data` with move semantics.

Parameters

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

9.25.2.4 TableField::TableField (const TableField & field)

Copy constructor.

Parameters

<i>field</i>	The field from which to copy.
--------------	-------------------------------

9.25.2.5 TableField::TableField (TableField && field)

Move constructor.

Parameters

<i>field</i>	The field from which to move.
--------------	-------------------------------

9.25.2.6 `TableField::~~TableField ()`

Destructor

9.25.3 Member Function Documentation

9.25.3.1 `size_t TableField::length () const`

Returns the length of the field.

Returns

The length of the field.

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

Overridden conversion operator.

Returns the field contents as a string.

9.25.3.3 `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.

9.25.3.4 `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.

9.25.3.5 `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.

9.25.3.6 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.

9.25.3.7 TableField & TableField::operator= (std::string && *data*)

Overridden assignment operator for `std::string` with move semantics.

Parameters

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

Returns

A reference to the same field.

9.25.3.8 TableField & TableField::operator= (const TableField & *field*)

Overridden copy assignment operator.

Parameters

<i>field</i>	The field to copy.
--------------	--------------------

Returns

A reference to the same field.

9.25.3.9 TableField & TableField::operator= (TableField && *field*)

Overridden move assignment operator.

Parameters

<i>field</i>	The field to move.
--------------	--------------------

Returns

A reference to the same field.

9.25.3.10 `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.

9.25.3.11 `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.

9.25.4 Friends And Related Function Documentation

9.25.4.1 `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`.

9.25.5 Member Data Documentation

9.25.5.1 `std::string glldb::TableField::m_data` `[private]`

The field contents

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

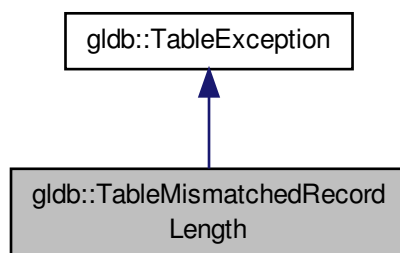
- lib/database/[tablefield.h](#)
- lib/database/[tablefield.cpp](#)

9.26 glldb::TableMismatchedRecordLength Class Reference

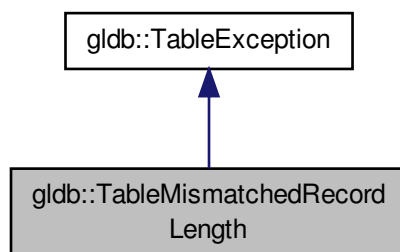
Mismatched record length exception class.

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

Inheritance diagram for glldb::TableMismatchedRecordLength:



Collaboration diagram for glldb::TableMismatchedRecordLength:



Public Member Functions

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

9.26.1 Detailed Description

Mismatched record length exception class.

9.26.2 Constructor & Destructor Documentation

9.26.2.1 `gldb::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:

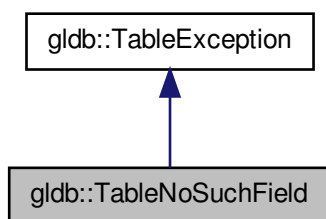
- [lib/database/table.h](#)

9.27 glDb::TableNoSuchField Class Reference

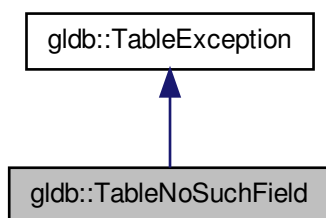
No such field exception class.

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

Inheritance diagram for `gldb::TableNoSuchField`:



Collaboration diagram for `gldb::TableNoSuchField`:



Public Member Functions

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

9.27.1 Detailed Description

No such field exception class.

9.27.2 Constructor & Destructor Documentation

9.27.2.1 `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:

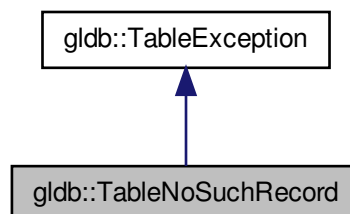
- lib/database/[table.h](#)

9.28 glldb::TableNoSuchRecord Class Reference

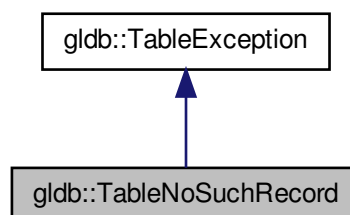
No such record exception class.

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

Inheritance diagram for `glldb::TableNoSuchRecord`:



Collaboration diagram for `glldb::TableNoSuchRecord`:



Public Member Functions

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

9.28.1 Detailed Description

No such record exception class.

9.28.2 Constructor & Destructor Documentation

9.28.2.1 `gldb::TableNoSuchRecord::TableNoSuchRecord (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](#)

9.29 glldb::TableRow Class Reference

Database table row class.

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

Public Member Functions

- [TableRow](#) ()
Constructor with initial number of fields.
- [TableRow](#) (const size_t [size](#))
Constructor with string vector.
- [TableRow](#) (const std::vector< std::string > &vec)
Constructor with string vector and move semantics.
- [TableRow](#) (std::vector< std::string > &&vec)
Copy constructor.
- [TableRow](#) (const [TableRow](#) &row)
Move constructor.
- [TableRow](#) ([TableRow](#) &&row)
Copy assignment operator.
- [TableRow](#) & [operator=](#) (const [TableRow](#) &row)
Move assignment operator.
- [~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](#) (std::string &&new_field)
Appends a field to the row with move semantics.
- void [append_field](#) (const [TableField](#) &new_field)
Appends a field to the row.
- void [append_field](#) ([TableField](#) &&new_field)
Appends a field to the row with move semantics.
- void [print](#) (std::ostream &stream) const
Prints a row.
- std::string [record_string](#) (const std::vector< bool > "ed) const
Creates a comma separated string of fields.
- std::string [record_string](#) () const
Creates an unquoted comma separated string of fields.

Private Attributes

- std::vector< [TableField](#) > [m_fields](#)

9.29.1 Detailed Description

Database table row class.

9.29.2 Constructor & Destructor Documentation

9.29.2.1 TableRow::TableRow ()

Default constructor

9.29.2.2 TableRow::TableRow (const size_t size) [explicit]

Constructor with initial number of fields.

Parameters

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

9.29.2.3 TableRow::TableRow (const std::vector< std::string > & vec) [explicit]

Constructor with string vector.

Parameters

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

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

Constructor with string vector and move semantics.

Parameters

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

9.29.2.5 TableRow::TableRow (const TableRow & *row*)

Copy constructor.

Parameters

<i>row</i>	The row to copy.
------------	------------------

9.29.2.6 TableRow::TableRow (TableRow && *row*) [explicit]

Move constructor.

Parameters

<i>row</i>	The row to move.
------------	------------------

9.29.2.7 TableRow::~~TableRow ()

Destructor

9.29.3 Member Function Documentation

9.29.3.1 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.
------------------	--------------------------------

9.29.3.2 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.
------------------	--------------------------------

9.29.3.3 void TableRow::append_field (std::string && *new_field*)

Appends a field to the row with move semantics.

Parameters

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

9.29.3.4 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.
------------------	-----------------------------

9.29.3.5 void TableRow::append_field (TableField && *new_field*)

Appends a field to the row with move semantics.

Parameters

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

9.29.3.6 TableRow & TableRow::operator= (const TableRow & *row*)

Copy assignment operator.

Parameters

<i>row</i>	The row to copy.
------------	------------------

Returns

A reference to the assigned-to row.

9.29.3.7 TableRow & TableRow::operator= (TableRow && *row*)

Move assignment operator.

Parameters

<i>row</i>	The row to move.
------------	------------------

Returns

A reference to the assigned-to row.

9.29.3.8 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.

9.29.3.9 `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.

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

Prints a row.

Parameters

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

9.29.3.11 `std::string TableRow::record_string (const std::vector< bool > & quoted) const`

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.

9.29.3.12 `std::string TableRow::record_string () const`

Creates an unquoted comma separated string of fields.

Returns

The unquoted comma separated string.

9.29.3.13 `size_t TableRow::size () const`

Returns the number of fields.

Returns

The number of fields.

9.29.4 Member Data Documentation

9.29.4.1 `std::vector<TableRow> 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](#)

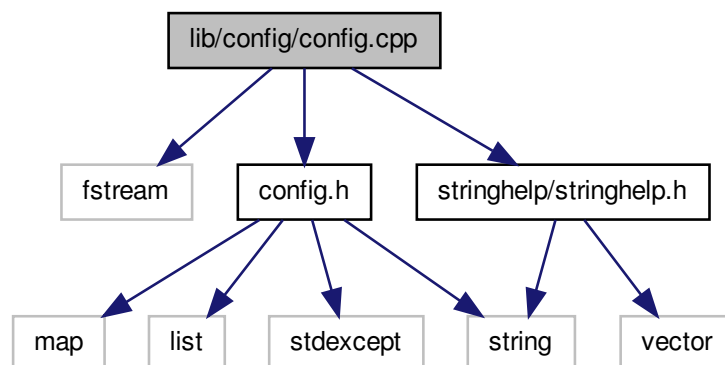
Chapter 10

File Documentation

10.1 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:
```



10.1.1 Detailed Description

Implementation of program configurations class.

Author

Paul Griffiths

Copyright

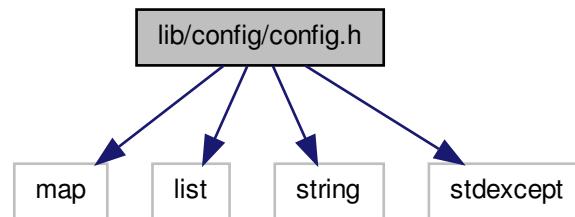
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.2 lib/config/config.h File Reference

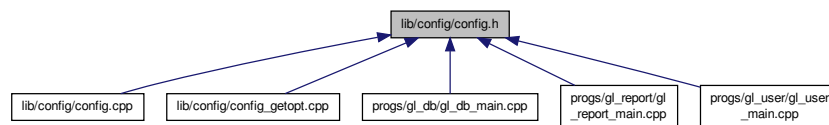
Interface to program configurations class.

```
#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.

10.2.1 Detailed Description

Interface to program configurations class.

Author

Paul Griffiths

Copyright

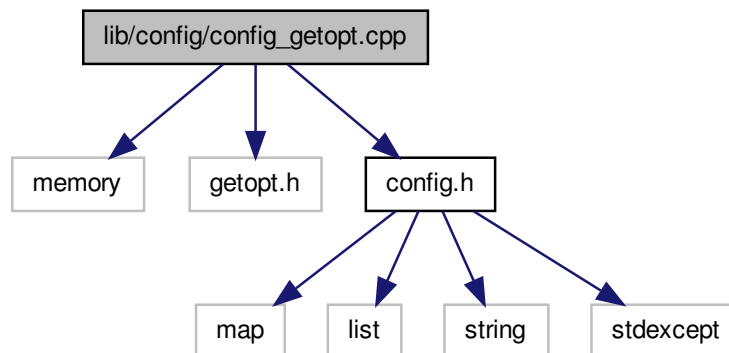
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.3 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`

10.3.1 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/>

10.3.2 Macro Definition Documentation

10.3.2.1 `#define _XOPEN_SOURCE 600`

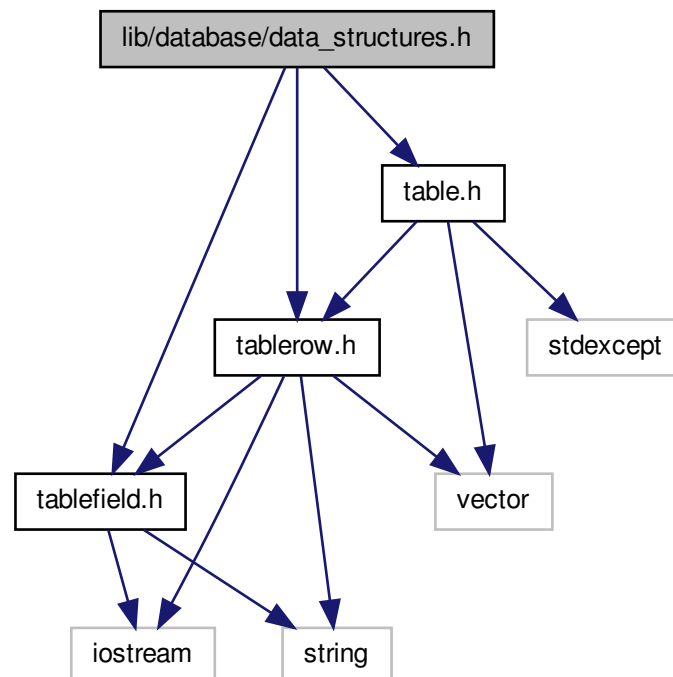
UNIX feature test macro for getopt library

10.4 `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`:





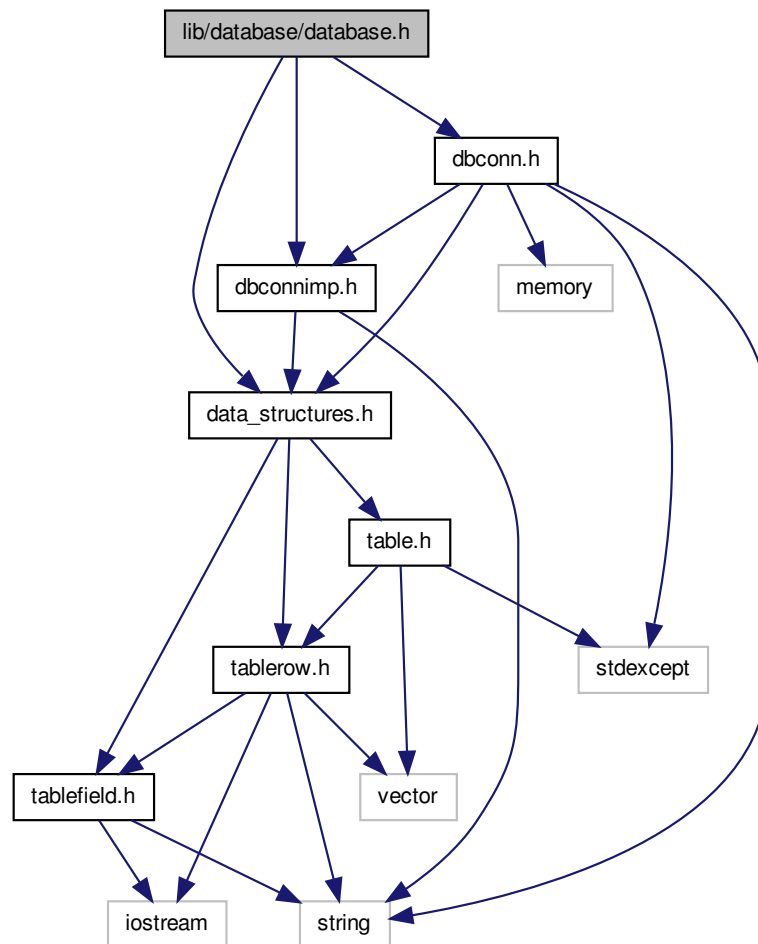
Author

Copyright

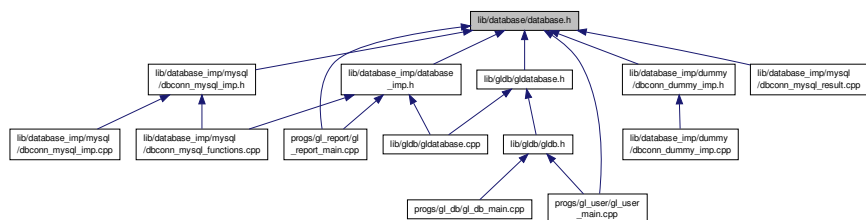
10.5 lib/database/database.h File Reference

```
#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:



10.5.1 Detailed Description

User interface to database functionality.

Author

Paul Griffiths

Copyright

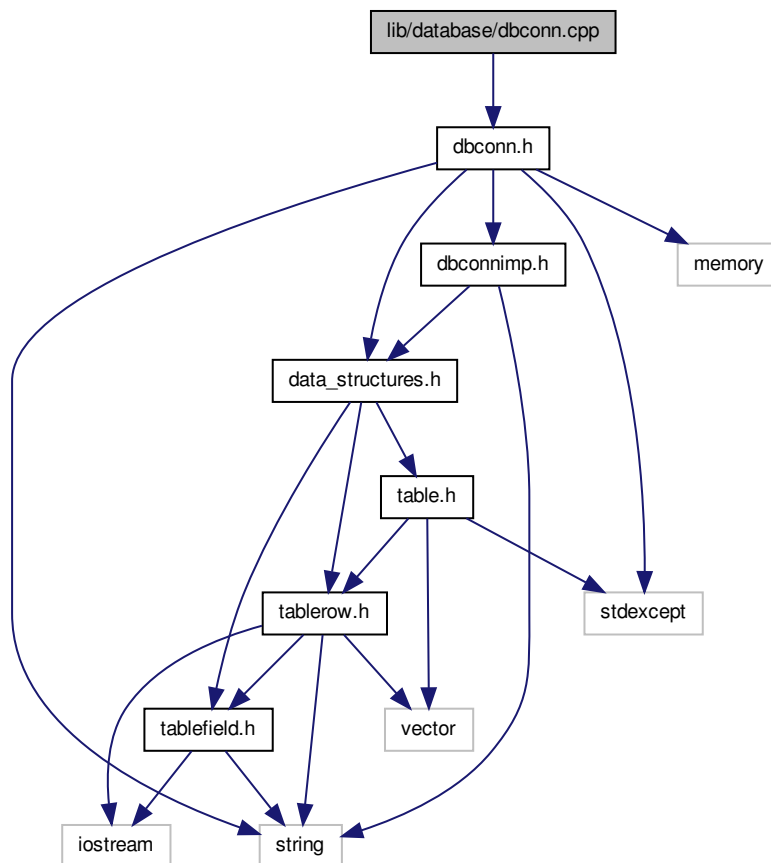
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.6 lib/database/dbconn.cpp File Reference

Implementation of database connection class.

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

Include dependency graph for dbconn.cpp:



10.6.1 Detailed Description

Implementation of database connection class.

Author

Paul Griffiths

Copyright

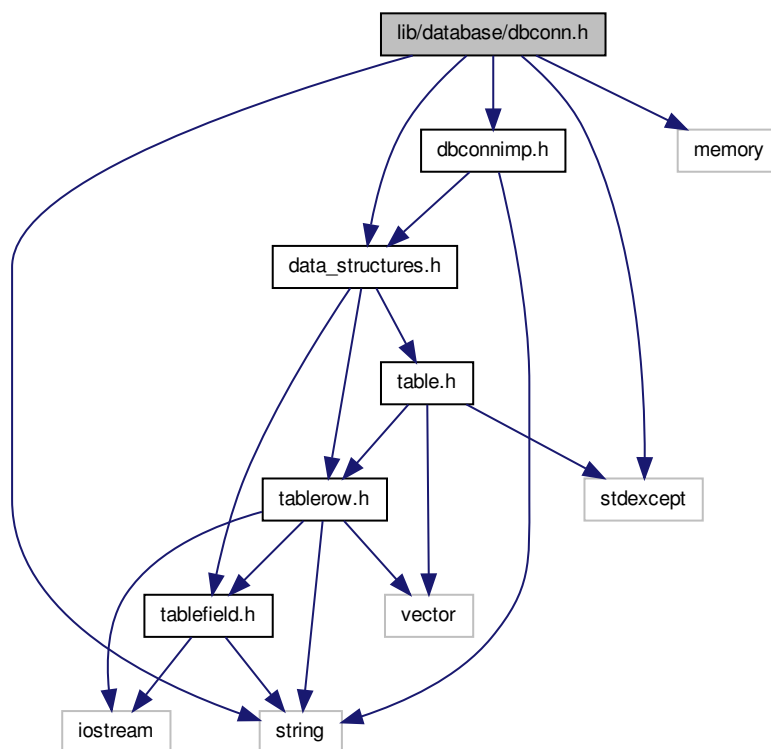
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.7 lib/database/dbconn.h File Reference

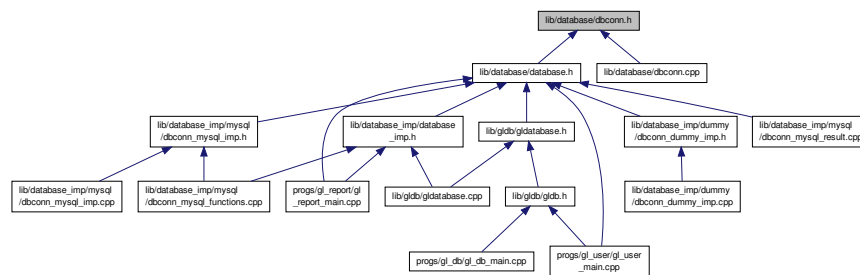
Interface to database connection base class.

```
#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.

10.7.1 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/>

10.8 lib/database/dbconnimp.h File Reference

Interface to abstract database implementation base class.

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


10.8.1 Detailed Description

Interface to abstract database implementation base class.

Author

Paul Griffiths

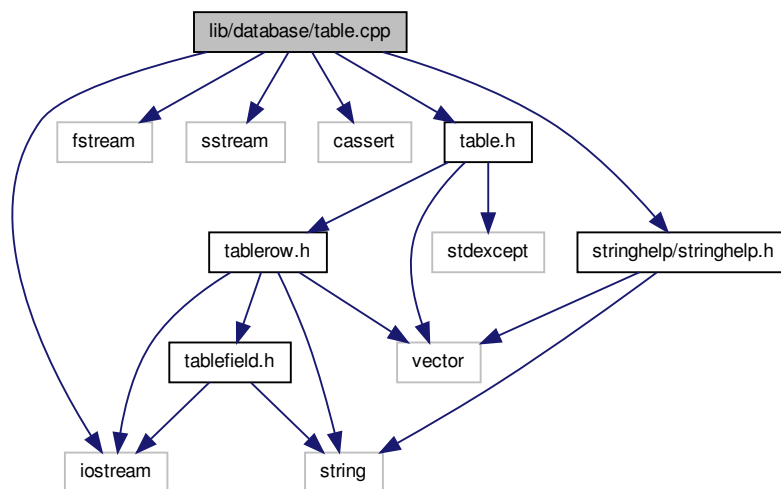
Copyright

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

10.9 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:
```



10.9.1 Detailed Description

Implementation of database table data structure.

Author

Paul Griffiths

Copyright

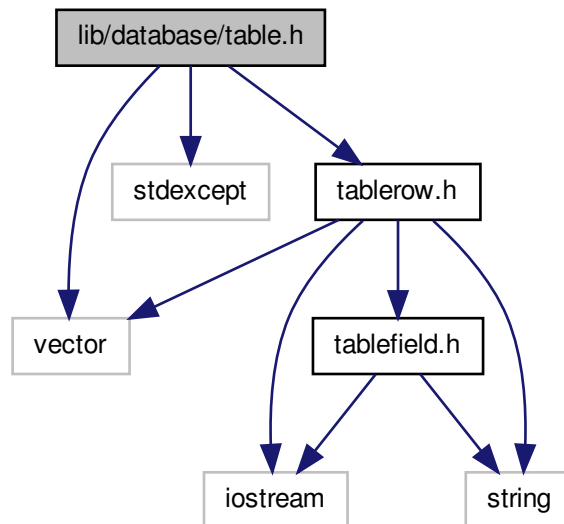
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.10 lib/database/table.h File Reference

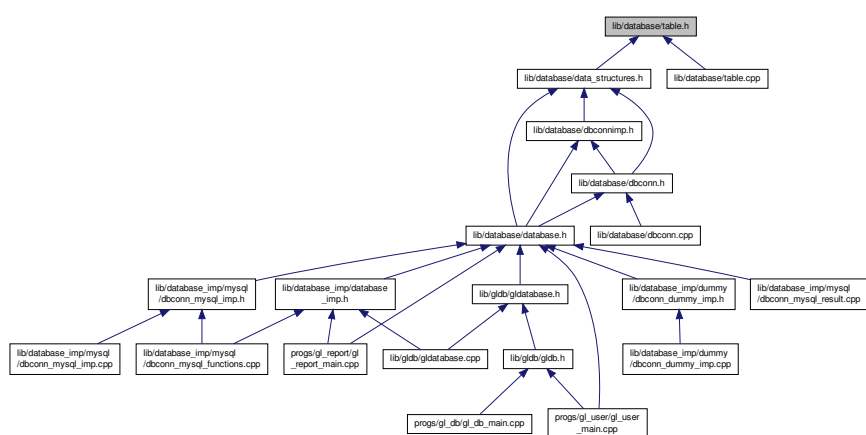
Interface to database table data structure.

```
#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.

10.10.1 Detailed Description

Interface to database table data structure.

Author

Paul Griffiths

Copyright

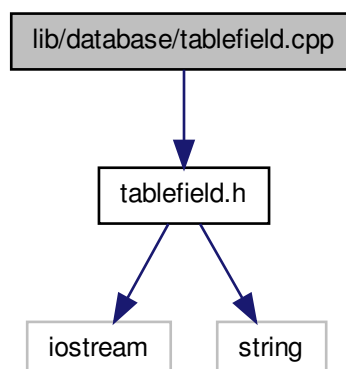
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.11 lib/database/tablefield.cpp File Reference

Implementation of database table field class.

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

Include dependency graph for tablefield.cpp:



Classes

- class `gldb::TableField`
Database table field class.

Functions

- `std::ostream & gldb::operator<< (std::ostream &out, const TableField &field)`
Overridden << operator for printing a field.

10.12.1 Detailed Description

Interface to database table field class.

Author

Paul Griffiths

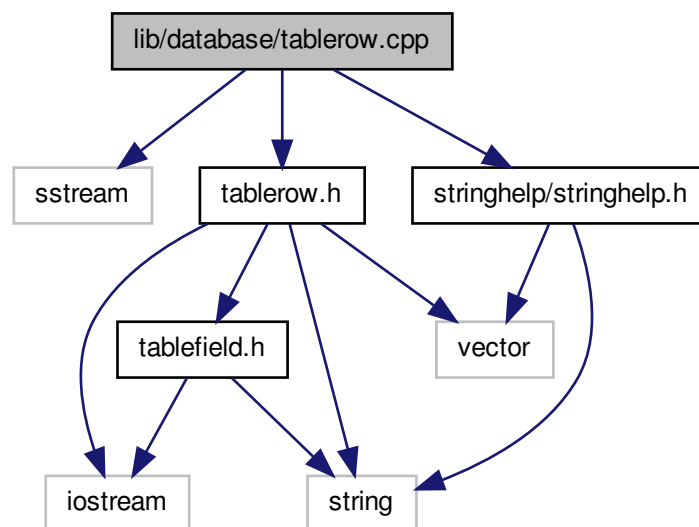
Copyright

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

10.13 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:
```



10.13.1 Detailed Description

Implementation of database table row data structure.

Author

Paul Griffiths

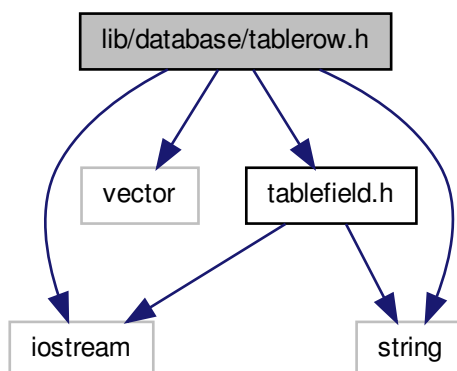
Copyright

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

10.14 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:
```



Copyright

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

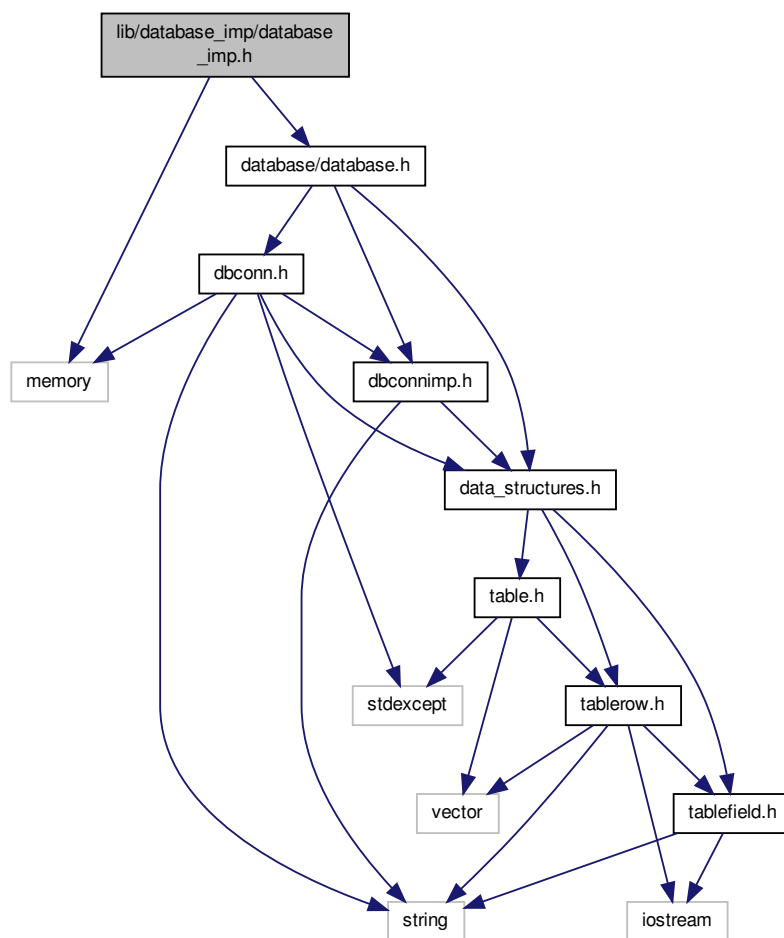
10.15 lib/database_imp/database_imp.h File Reference

Interface to database implementation factory function.

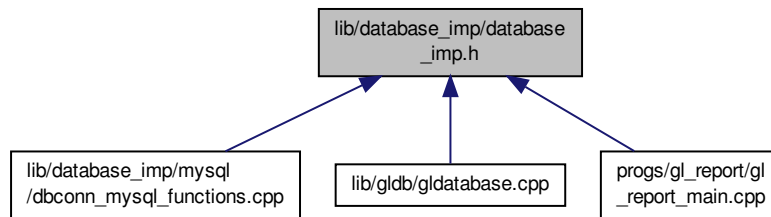
```
#include <memory>
```

```
#include "database/database.h"
```

Include dependency graph for database_imp.h:



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



Functions

- `DBConnImp * 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.

10.15.1 Detailed Description

Interface to database implementation factory function.

Author

Paul Griffiths

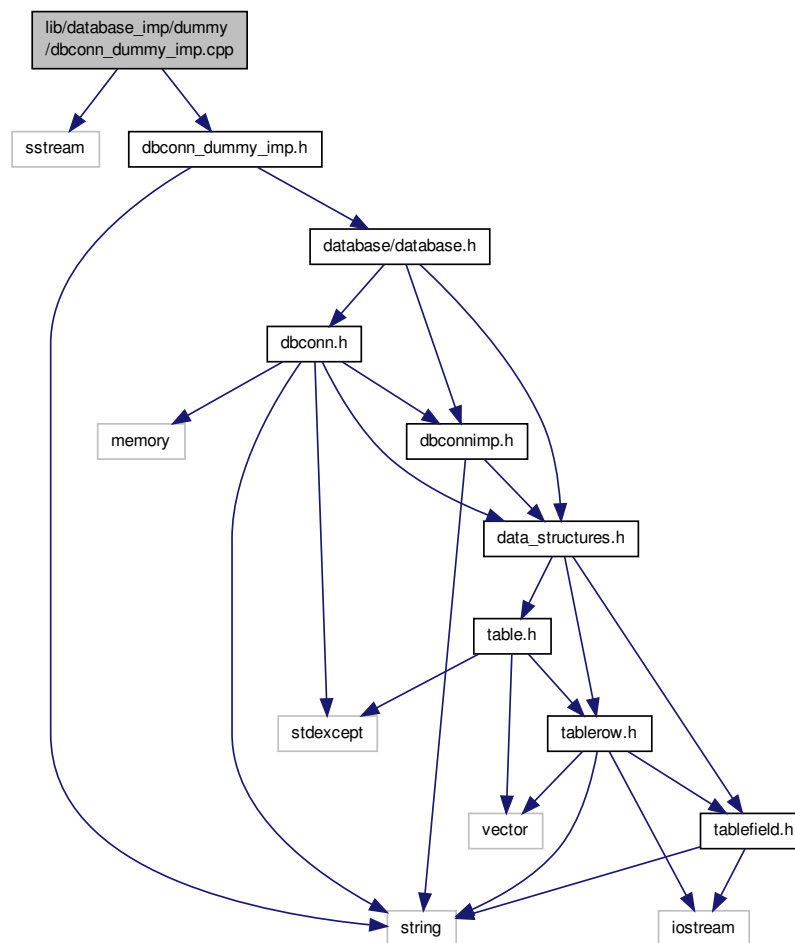
Copyright

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

10.16 lib/database_imp/dummy/dbconn_dummy_imp.cpp File Reference

Implementation of Dummy database connection implementation class.

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



10.16.1 Detailed Description

Implementation of Dummy database connection implementation class.

Author

Paul Griffiths

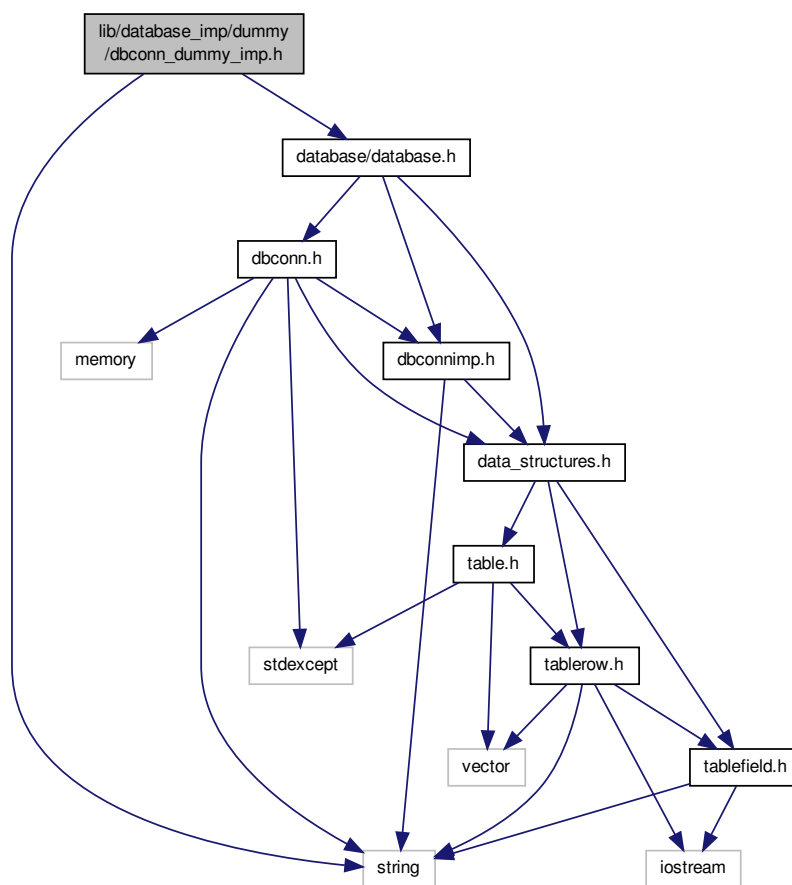
Copyright

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

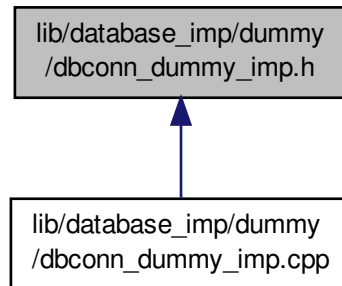
10.17 lib/database_imp/dummy/dbconn_dummy_imp.h File Reference

Interface to dummy database connection implementation class.

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



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



Classes

- class [gldb::DBConnDummy](#)

Dummy database implementation class.

10.17.1 Detailed Description

Interface to dummy database connection implementation class.

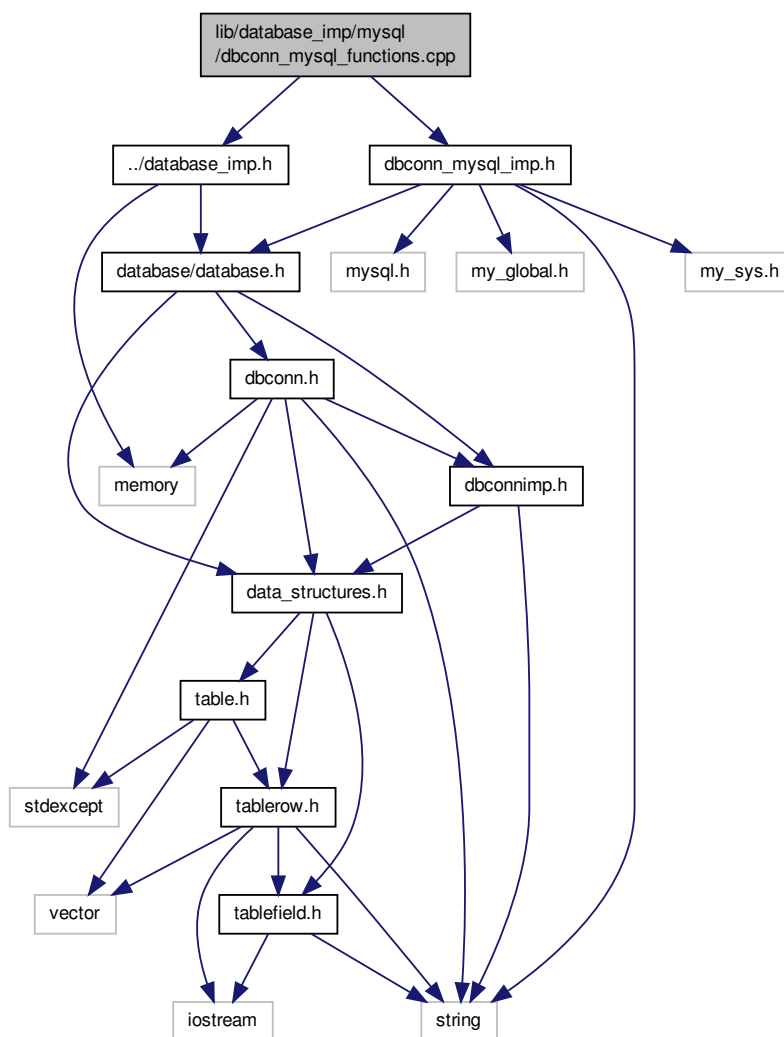
Author

Paul Griffiths

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

Implementation of MySQL implementation factory function.

```
#include "../database_imp.h"
#include "dbconn_mysql_imp.h"
Include dependency graph for dbconn_mysql_functions.cpp:
```



Implementation of MySQL 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/>

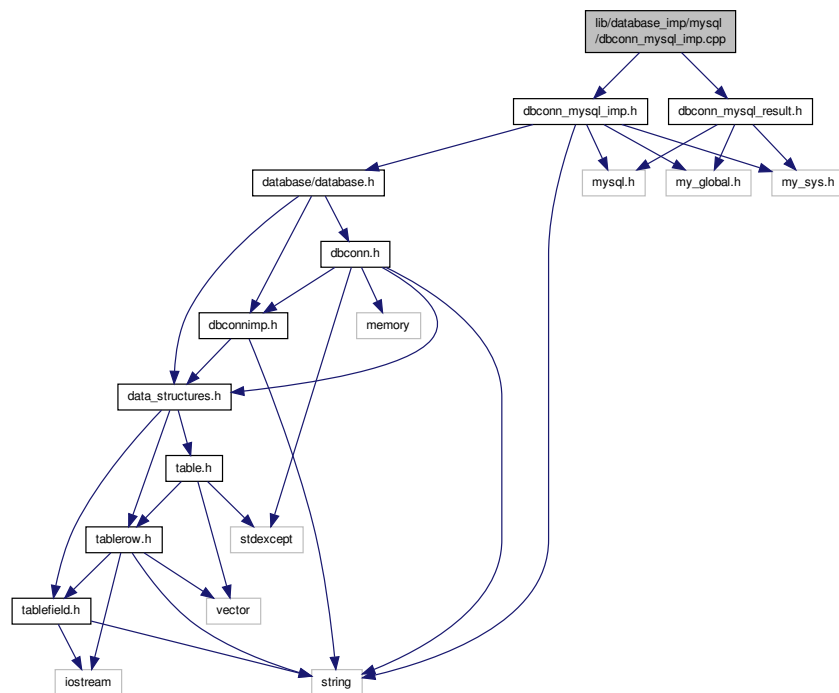
10.19 lib/database_imp/mysql/dbconn_mysql_imp.cpp File Reference

Implementation of MySQL database connection implementation class.

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

```
#include "dbconn_mysql_result.h"
```

Include dependency graph for dbconn_mysql_imp.cpp:

**Functions**

- static `TableRow get_field_names (MySQLResult &result)`
Gets field names from a MySQL result structure.
- static `TableRow get_row (MySQLResult &result, MYSQL_ROW row)`
Creates a TableRow from a MySQL result row.

10.19.1 Detailed Description

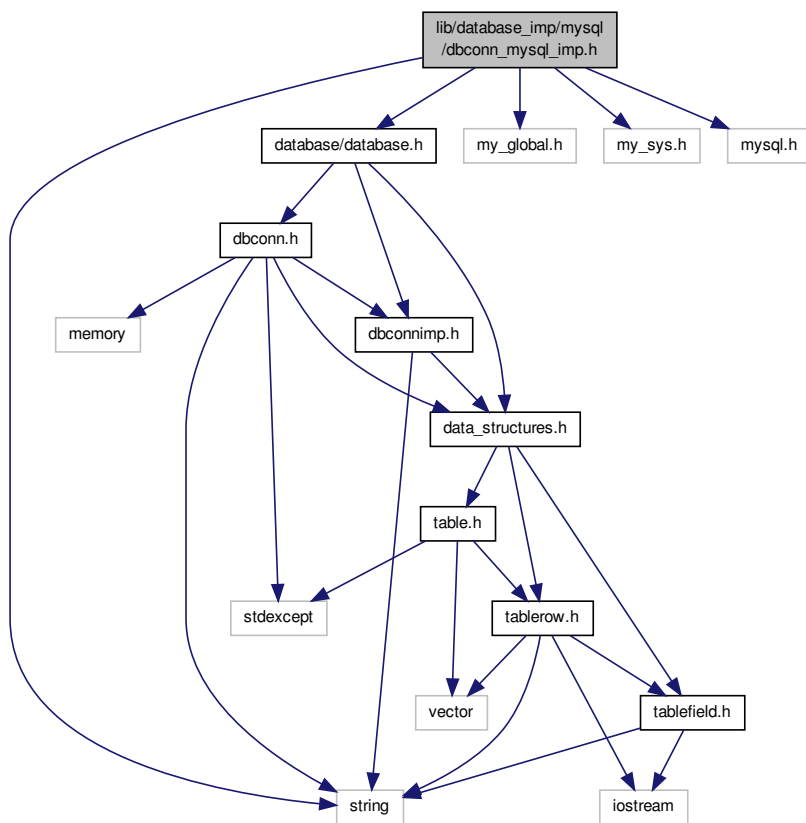
Implementation of MySQL database connection implementation class.

Paul Griffiths

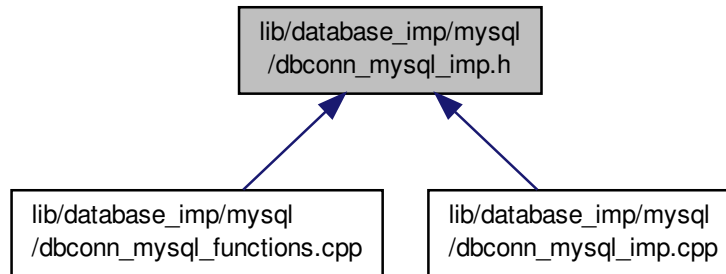
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

```
#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.

10.20.1 Detailed Description

Interface to MySQL database connection implementation class.

Author

Paul Griffiths

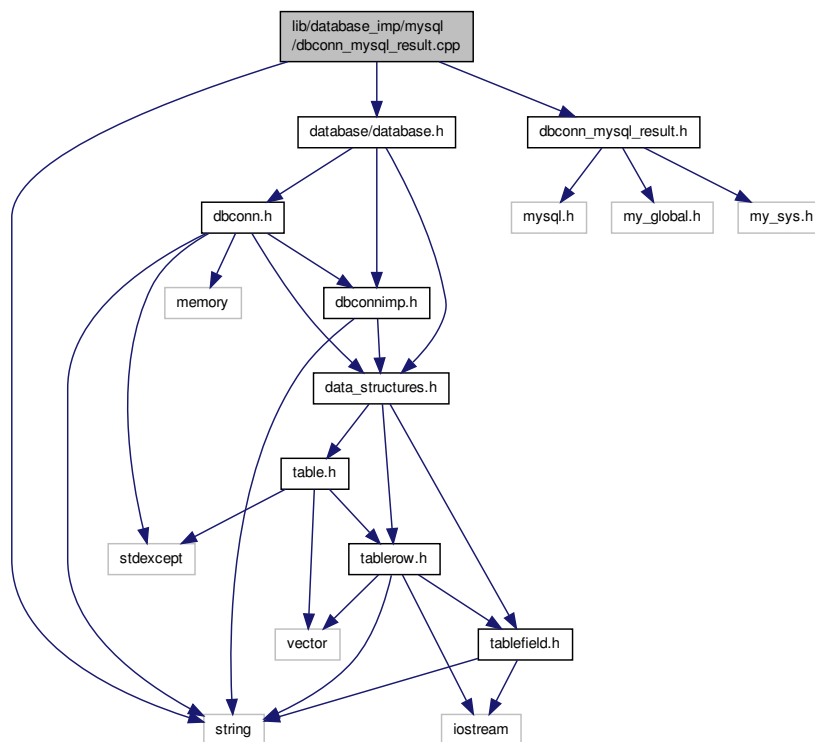
Copyright

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

10.21 lib/database_imp/mysql/dbconn_mysql_result.cpp File Reference

Implementation of MySQL result structure resource handle class.

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



10.21.1 Detailed Description

Implementation of MySQL result structure resource handle class.

Author

Paul Griffiths

Copyright

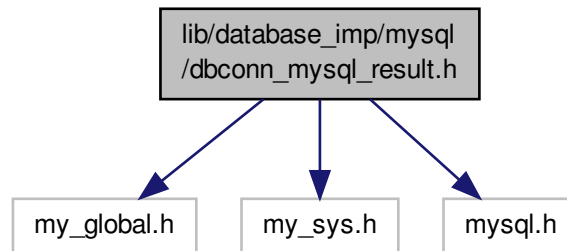
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.22 lib/database_imp/mysql/dbconn_mysql_result.h File Reference

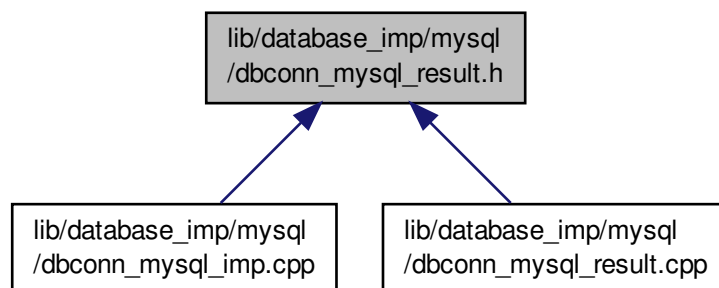
Interface to MySQL result structure resource handle class.

```
#include <my_global.h>
#include <my_sys.h>
#include <mysql.h>
```

Include dependency graph for dbconn_mysql_result.h:



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



Classes

- class [glldb::MySQLResult](#)
MySQL result structure class.

10.22.1 Detailed Description

Interface to MySQL result structure resource handle class.

Author

Paul Griffiths

Copyright

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

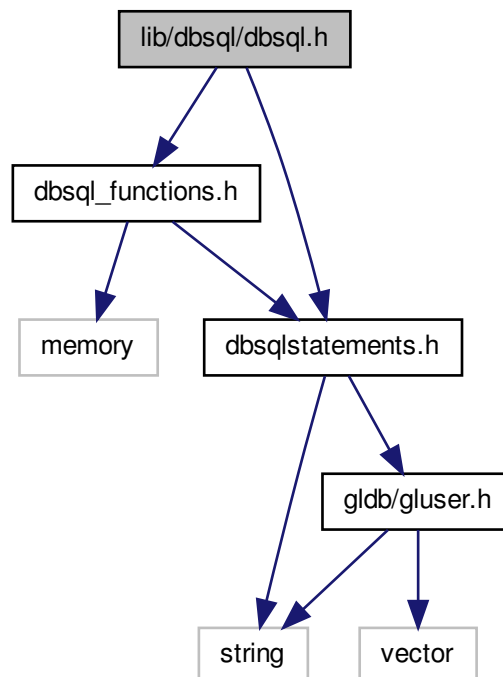
10.23 lib/dbsql/dbsql.h File Reference

User interface to DBSQL module.

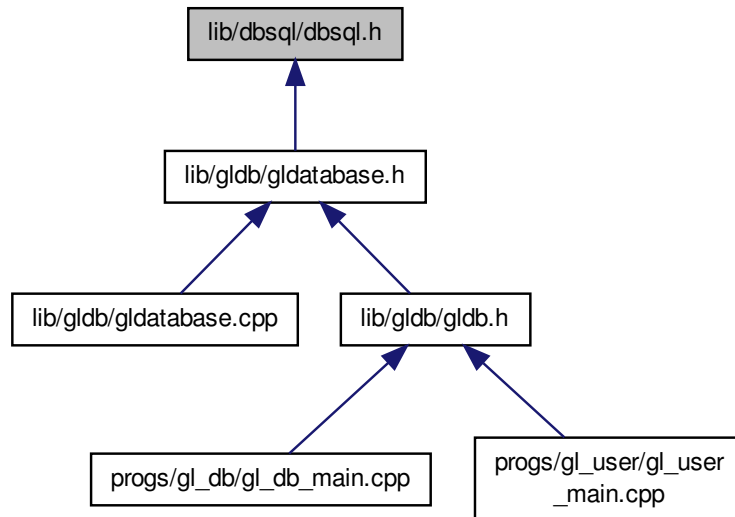
```
#include "dbsql_functions.h"
```

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

Include dependency graph for dbsql.h:



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



10.23.1 Detailed Description

User interface to DBSQL module.

Author

Paul Griffiths

Copyright

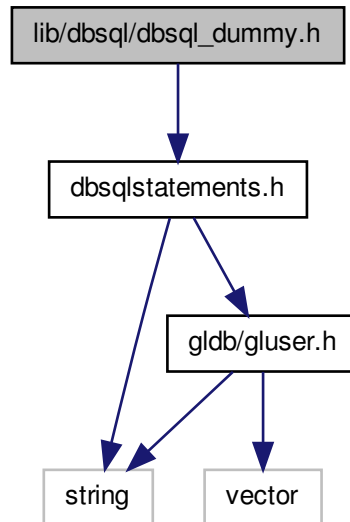
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.24 lib/dbsql/dbsql_dummy.h File Reference

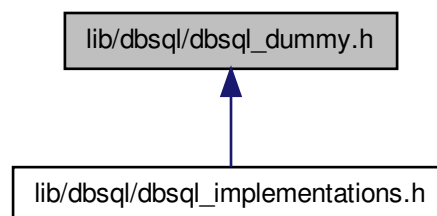
Interface to dummy SQL statement class.

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

Include dependency graph for `dbsql_dummy.h`:



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



Classes

- class `genleg::DBSQLDummy`
Dummy SQL statements class.

10.24.1 Detailed Description

Interface to dummy SQL statement class.

Author

Paul Griffiths

Copyright

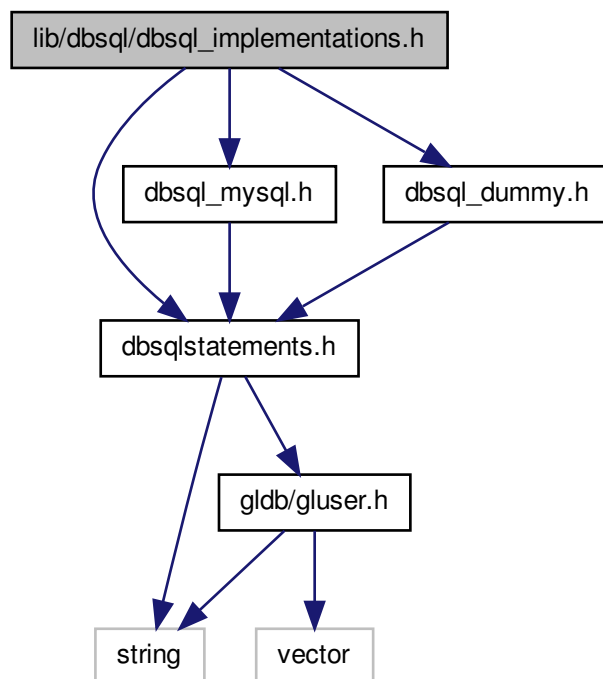
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.25 lib/dbsql/dbsql_implementations.h File Reference

Aggregation header for DBSqlStatements implementations.

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

Include dependency graph for dbsql_implementations.h:



10.25.1 Detailed Description

Aggregation header for DBSqlStatements implementations.

Author

Paul Griffiths

Copyright

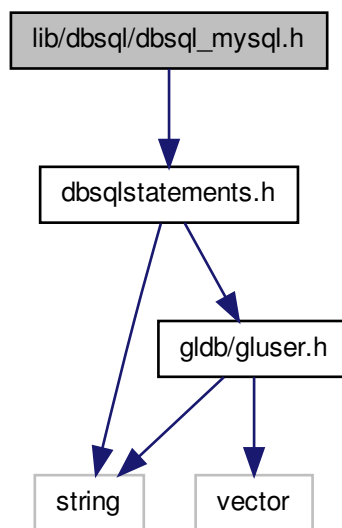
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.26 lib/dbsql/dbsql_mysql.h File Reference

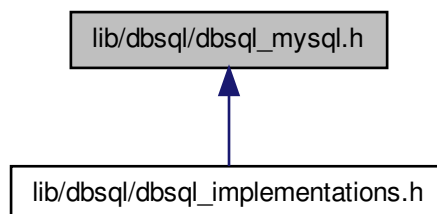
Interface to MySQL SQL statement class.

```
#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.

10.26.1 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/>

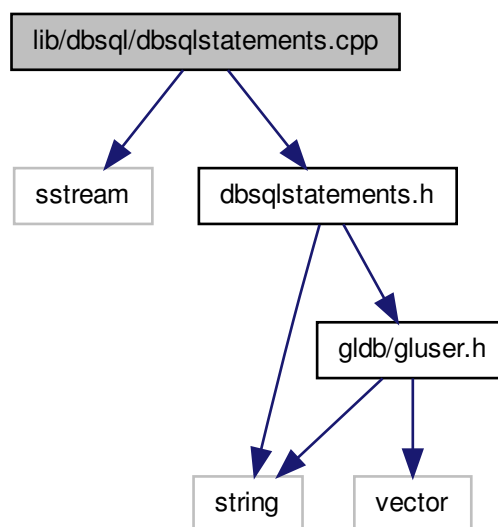
10.27 lib/dbsql/dbsqlstatements.cpp File Reference

Implementation of SQL statement class.

```
#include <sstream>
```

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

Include dependency graph for dbsqlstatements.cpp:



10.27.1 Detailed Description

Implementation of SQL statement class.

Author

Paul Griffiths

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

Implementation of SQL module standalone functions.

```
graph TD; A[lib/dbsql/dbsqlstatements.h] --> B[gdb/gdbuser.h]; A --> C[string]; B --> C; B --> D[vector];
```

The diagram illustrates the relationship between the `lib/dbsql/dbsqlstatements.h` header file and other components. It shows that `lib/dbsql/dbsqlstatements.h` includes `gdb/gdbuser.h` and `string`. Additionally, `gdb/gdbuser.h` includes both `string` and `vector`.

[illegible]

- class `genleg::DBSQLStatements`

SQL statements class.

10.28.1 Detailed Description

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

Interface to SQL module standalone functions.

Author

Paul Griffiths

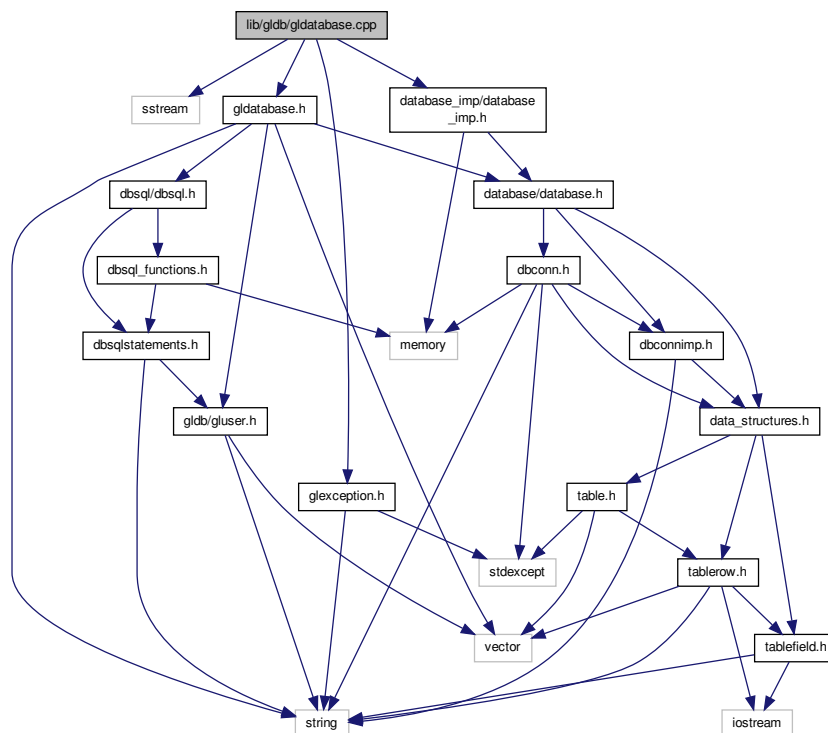
Copyright

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

10.29 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" })

10.29.1 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/>

10.29.2 Function Documentation

10.29.2.1 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

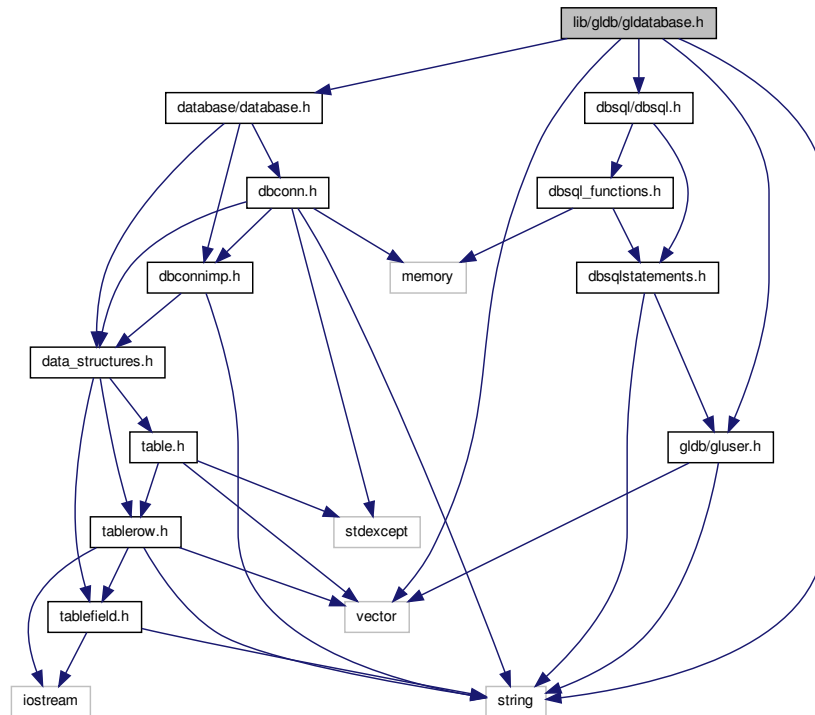
<i>GLDBException</i>	if <i>bs</i> contains any other value.
----------------------	--

10.30 lib/gldb/gldatabase.h File Reference

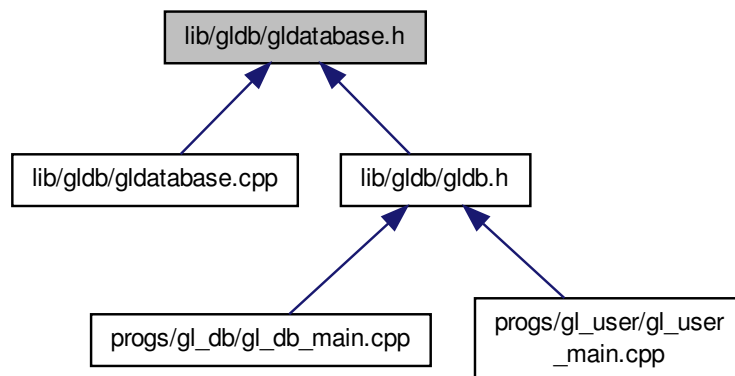
Interface to General Ledger database class.

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

Include dependency graph for `gldatabase.h`:



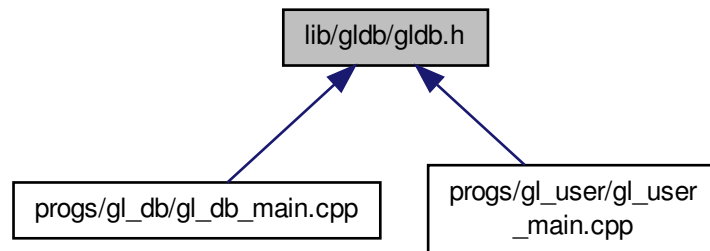
This graph shows which files directly or indirectly include this file:



Classes

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

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



10.31.1 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/>

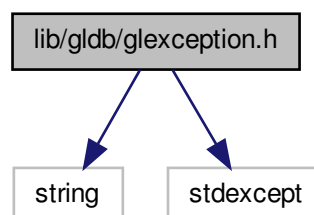
10.32 lib/gldb/glexception.h File Reference

Interface to General Ledger base exception class.

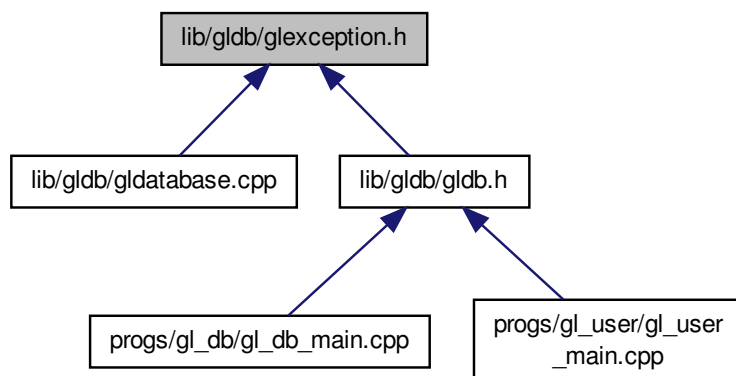
```
#include <string>
```

```
#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 exceptionc class.

10.32.1 Detailed Description

Interface to General Ledger base exception class.

Author

Paul Griffiths

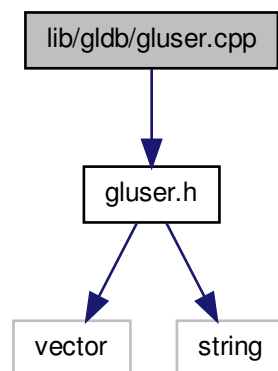
Copyright

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

10.33 lib/gdb/gluser.cpp File Reference

Implementation of user class.

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



10.33.1 Detailed Description

Implementation of user class.

Author

Paul Griffiths

Copyright

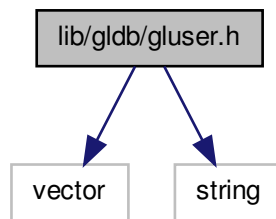
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.34 lib/gdb/gluser.h File Reference

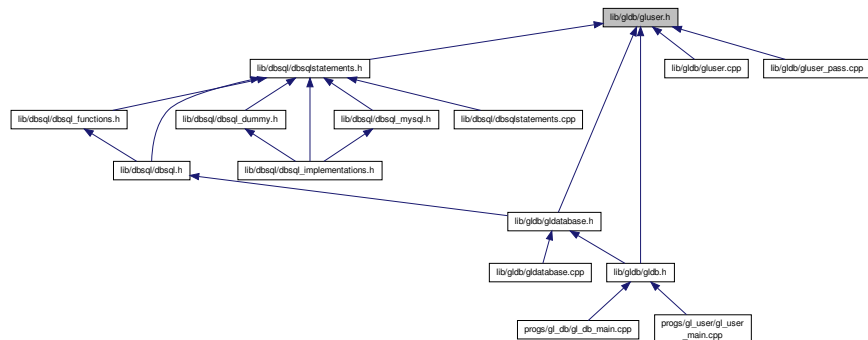
Interface to user class.

```
#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.

10.34.1 Detailed Description

Interface to user class.

Author

Paul Griffiths

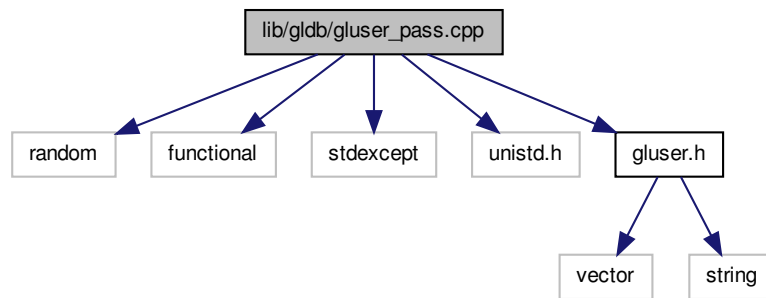
Copyright

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

10.35 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()

10.35.1 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/>

10.35.2 Macro Definition Documentation

10.35.2.1 `#define _XOPEN_SOURCE 600`

UNIX feature test macro

10.35.3 Function Documentation

10.35.3.1 static std::string generate_salt () [static]

Generates a random two-character salt for crypt()

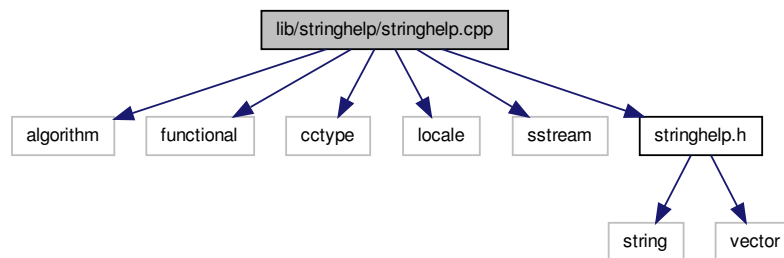
Returns

The two-character salt.

10.36 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:
```



10.36.1 Detailed Description

Implementation of string helper functions.

Author

Paul Griffiths

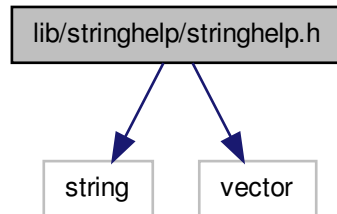
Copyright

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

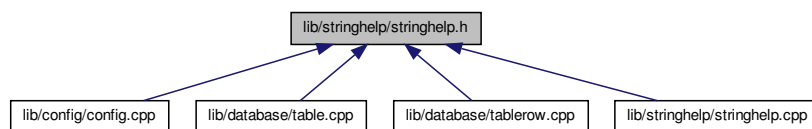
10.37 lib/stringhelp/stringhelp.h File Reference

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 (const 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.

10.37.1 Detailed Description

Interface to string helper functions.

Author

Paul Griffiths

Copyright

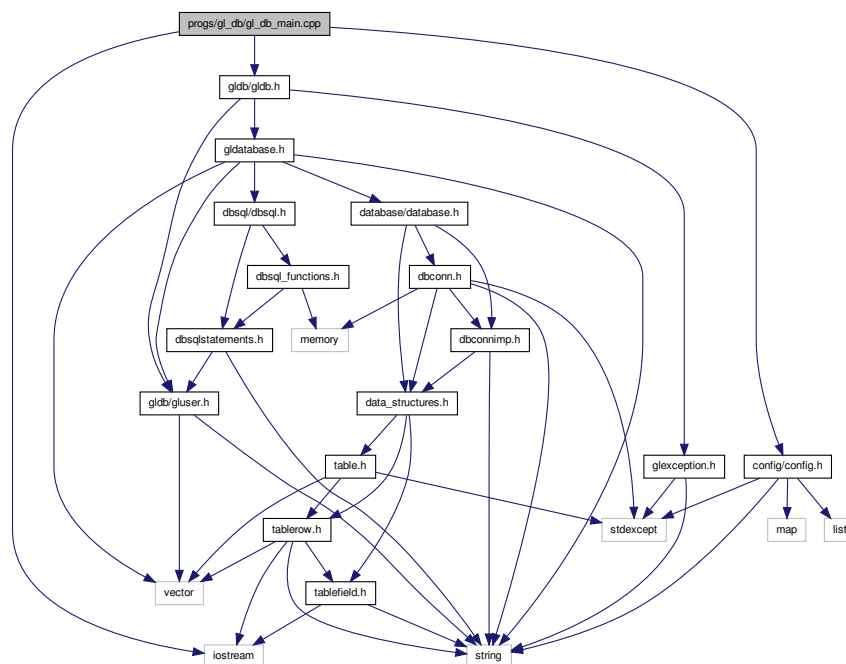
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.38 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 * `progrname = "gl_db"`
Static variable for program name.

10.38.1 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/>

10.39 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"
```


10.39.1 Detailed Description

Main functionality for gl_report program.

Author

Paul Griffiths

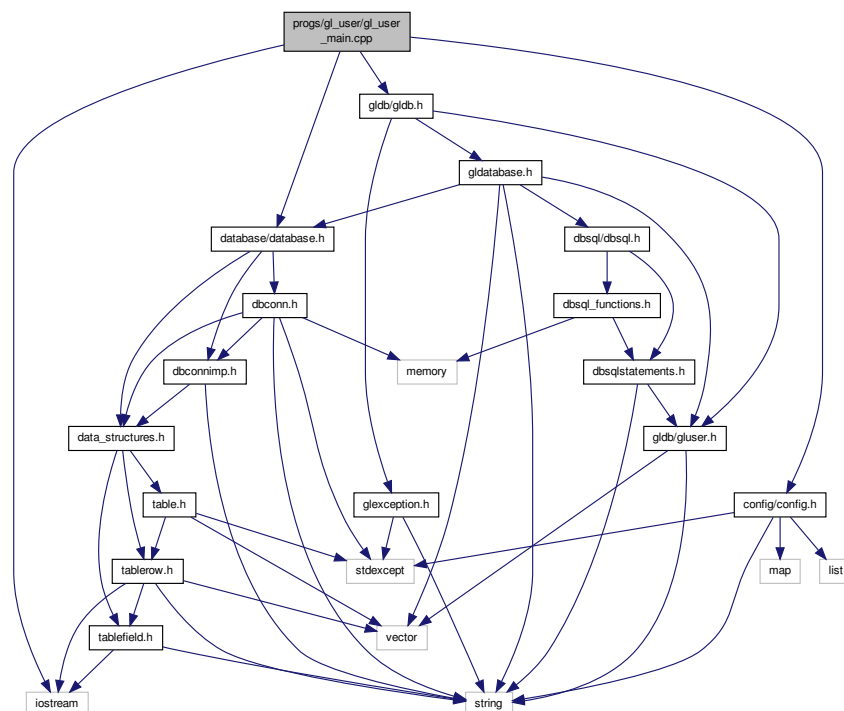
Copyright

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

10.40 progs/gl_user/gl_user_main.cpp File Reference

Main functionality for gl_user program.

```
#include <iostream>
#include "gldb/gldb.h"
#include "database/database.h"
#include "config/config.h"
Include dependency graph for gl_user_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.
- `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 * `programe` = "gl_user"
Static variable for program name.

10.40.1 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/>

10.40.2 Function Documentation

10.40.2.1 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.

10.40.2.2 `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.

10.40.2.3 `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.

10.40.2.4 `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.

10.40.2.5 `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.

10.40.2.6 `static std::string login (void) [static]`

Gets a password from the terminal.

Returns

The password.

10.40.2.7 `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.

10.40.2.8 `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>	<i>argc</i> passed to main() .
<i>argv</i>	<i>argv</i> passed to main() .

10.40.2.9 `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.

10.40.2.10 `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, [27](#)
- ~DBConnDummy
 - gldb::DBConnDummy, [41](#)
- ~DBConnImp
 - gldb::DBConnImp, [43](#)
- ~DBConnMySQL
 - gldb::DBConnMySQL, [46](#)
- ~DBSQLStatements
 - genleg::DBSQLStatements, [50](#)
- ~GLDatabase
 - genleg::GLDatabase, [54](#)
- ~GLUser
 - genleg::GLUser, [59](#)
- ~MySQLResult
 - gldb::MySQLResult, [63](#)
- ~Table
 - gldb::Table, [66](#)
- ~TableField
 - gldb::TableField, [76](#)
- ~TableRow
 - gldb::TableRow, [84](#)
- _XOPEN_SOURCE
 - config_getopt.cpp, [92](#)
 - gluser_pass.cpp, [132](#)
- add_cmdline_option
 - genleg::Config, [28](#)
- append_field
 - gldb::TableRow, [84](#), [85](#)
- append_record
 - gldb::Table, [67](#)
- backend
 - genleg::GLDatabase, [54](#)
- boolstring_to_bool
 - gldatabase.cpp, [125](#)
- check_db_parameters
 - Database program., [25](#)
 - gl_user_main.cpp, [139](#)
- check_help_and_version
 - Database program., [25](#)
 - gl_user_main.cpp, [140](#)
- check_password
 - genleg::GLUser, [59](#)
- check_user_password
 - gl_user_main.cpp, [140](#)
- Config
 - genleg::Config, [27](#)
- config_getopt.cpp
 - _XOPEN_SOURCE, [92](#)
- ConfigBadConfigFile
 - genleg::ConfigBadConfigFile, [30](#)
- ConfigBadOption
 - genleg::ConfigBadOption, [31](#)
- ConfigCouldNotOpenFile
 - genleg::ConfigCouldNotOpenFile, [33](#)
- ConfigException
 - genleg::ConfigException, [34](#)
- ConfigOptionNotSet
 - genleg::ConfigOptionNotSet, [35](#)
- create_from_file
 - gldb::Table, [67](#)
- create_structure
 - genleg::GLDatabase, [54](#)
- create_table
 - genleg::DBSQLStatements, [50](#)
- create_user
 - genleg::GLDatabase, [55](#)
- create_view
 - genleg::DBSQLStatements, [50](#)
- DBConn
 - gldb::DBConn, [36](#)
- DBConnCouldNotConnect
 - gldb::DBConnCouldNotConnect, [38](#)
- DBConnCouldNotQuery
 - gldb::DBConnCouldNotQuery, [39](#)
- DBConnDummy
 - gldb::DBConnDummy, [41](#)
- DBConnException
 - gldb::DBConnException, [42](#)
- DBConnImp
 - gldb::DBConnImp, [43](#)
- DBConnMySQL
 - gldb::DBConnMySQL, [45](#), [46](#)
- DBSQLStatements
 - genleg::DBSQLStatements, [50](#)
- Database interaction module, [16](#)
 - get_connection, [17](#)
 - get_database_type, [17](#)
 - get_field_names, [17](#)
 - get_row, [17](#)
- Database program., [25](#)
 - check_db_parameters, [25](#)
 - check_help_and_version, [25](#)
 - login, [26](#)
 - main, [26](#)
 - set_configuration, [26](#)

- destroy_structure
 - genleg::GLDatabase, 55
- drop_table
 - genleg::DBSQLStatements, 50
- drop_view
 - genleg::DBSQLStatements, 50
- enable_user
 - gl_user_main.cpp, 140
- enabled
 - genleg::GLUser, 60
- firstname
 - genleg::GLUser, 60
- GLDBException
 - genleg::GLDBException, 57
- GLDatabase
 - genleg::GLDatabase, 54
- GLUser
 - genleg::GLUser, 59
- General Ledger database module., 15
- General purpose helpers., 21
 - split, 21
 - trim, 21
 - trim_back, 22
 - trim_front, 22
- generate_salt
 - gluser_pass.cpp, 133
- genleg::Config, 27
 - ~Config, 27
 - add_cmdline_option, 28
 - Config, 27
 - is_set, 28
 - m_opts_set, 29
 - m_opts_supp, 29
 - populate_from_cmdline, 28
 - populate_from_file, 29
- genleg::ConfigBadConfigFile, 29
 - ConfigBadConfigFile, 30
- genleg::ConfigBadOption, 31
 - ConfigBadOption, 31
- genleg::ConfigCouldNotOpenFile, 32
 - ConfigCouldNotOpenFile, 33
- genleg::ConfigException, 33
 - ConfigException, 34
- genleg::ConfigOptionNotSet, 34
 - ConfigOptionNotSet, 35
- genleg::DBSQLDummy, 47
- genleg::DBSQLMySQL, 48
- genleg::DBSQLStatements, 49
 - ~DBSQLStatements, 50
 - create_table, 50
 - create_view, 50
 - DBSQLStatements, 50
 - drop_table, 50
 - drop_view, 50
 - get_perms, 51
 - grant, 51
 - revoke, 51
 - update_user, 52
 - user_by_id, 52
 - user_by_username, 52
- genleg::GLDBException, 57
 - GLDBException, 57
- genleg::GLDatabase, 52
 - ~GLDatabase, 54
 - backend, 54
 - create_structure, 54
 - create_user, 55
 - destroy_structure, 55
 - GLDatabase, 54
 - get_user_by_id, 55
 - get_user_by_username, 55
 - grant, 56
 - load_sample_data, 56
 - m_dbc, 56
 - m_sql, 57
 - m_tables, 57
 - m_views, 57
 - revoke, 56
 - update_user, 56
- genleg::GLUser, 58
 - ~GLUser, 59
 - check_password, 59
 - enabled, 60
 - firstname, 60
 - GLUser, 59
 - id, 60
 - lastname, 60
 - m_enabled, 62
 - m_firstname, 62
 - m_id, 62
 - m_lastname, 62
 - m_pass_hash, 62
 - m_pass_salt, 62
 - m_perms, 62
 - m_username, 62
 - pass_hash, 60
 - pass_salt, 60
 - permissions, 61
 - set_enabled, 61
 - set_firstname, 61
 - set_lastname, 61
 - set_password, 61
 - set_username, 61
 - username, 62
- get_connection
 - Database interaction module, 17
- get_database_type
 - Database interaction module, 17
- get_field
 - gldb::Table, 67
- get_field_names
 - Database interaction module, 17
- get_headers
 - gldb::Table, 68

- get_perms
 - genleg::DBSQLStatements, 51
- get_row
 - Database interaction module, 17
- get_user
 - gl_user_main.cpp, 140
- get_user_by_id
 - genleg::GLDatabase, 55
- get_user_by_username
 - genleg::GLDatabase, 55
- gl_user_main.cpp
 - check_db_parameters, 139
 - check_help_and_version, 140
 - check_user_password, 140
 - enable_user, 140
 - get_user, 140
 - login, 140
 - main, 141
 - set_configuration, 141
 - set_user_password, 141
 - show_user_details, 141
- gldatabase.cpp
 - boolstring_to_bool, 125
- gldb::DBConn, 35
 - DBConn, 36
 - m_imp, 37
 - operator=, 36
 - query, 36
 - select, 37
- gldb::DBConnCouldNotConnect, 37
 - DBConnCouldNotConnect, 38
- gldb::DBConnCouldNotQuery, 38
 - DBConnCouldNotQuery, 39
- gldb::DBConnDummy, 40
 - ~DBConnDummy, 41
 - DBConnDummy, 41
 - operator=, 41
 - query, 41
 - select, 41
- gldb::DBConnException, 42
 - DBConnException, 42
- gldb::DBConnImp, 43
 - ~DBConnImp, 43
 - DBConnImp, 43
 - query, 44
 - select, 44
- gldb::DBConnMySQL, 44
 - ~DBConnMySQL, 46
 - DBConnMySQL, 45, 46
 - m_conn, 47
 - operator=, 46
 - query, 46
 - select, 46
- gldb::MySQLResult, 63
 - ~MySQLResult, 63
 - m_num_fields, 64
 - m_result, 64
 - MySQLResult, 63
 - num_fields, 64
 - operator=, 64
 - result, 64
- gldb::Table, 64
 - ~Table, 66
 - append_record, 67
 - create_from_file, 67
 - get_field, 67
 - get_headers, 68
 - insert_query, 68
 - m_headers, 69
 - m_quoted, 69
 - m_records, 70
 - num_fields, 68
 - num_records, 68
 - operator=, 68, 69
 - set_quoted, 69
 - Table, 66
- gldb::TableBadInputFile, 70
 - TableBadInputFile, 71
- gldb::TableCouldNotOpenInputFile, 71
 - TableCouldNotOpenInputFile, 72
- gldb::TableException, 72
 - TableException, 73
- gldb::TableField, 73
 - ~TableField, 76
 - length, 76
 - m_data, 78
 - operator std::string, 76
 - operator<<, 78
 - operator+=, 76
 - operator=, 76, 77
 - TableField, 75
- gldb::TableMismatchedRecordLength, 79
 - TableMismatchedRecordLength, 79
- gldb::TableNoSuchField, 80
 - TableNoSuchField, 81
- gldb::TableNoSuchRecord, 81
 - TableNoSuchRecord, 82
- gldb::TableRow, 82
 - ~TableRow, 84
 - append_field, 84, 85
 - m_fields, 87
 - operator=, 85
 - print, 86
 - record_string, 86
 - size, 86
 - TableRow, 83, 84
- gluser_pass.cpp
 - _XOPEN_SOURCE, 132
 - generate_salt, 133
- grant
 - genleg::DBSQLStatements, 51
 - genleg::GLDatabase, 56
- id
 - genleg::GLUser, 60
- insert_query
 - gldb::Table, 68

is_set
 genleg::Config, 28

 lastname
 genleg::GLUser, 60
 length
 gldb::TableField, 76
 lib/config/config.cpp, 89
 lib/config/config.h, 90
 lib/config/config_getopt.cpp, 91
 lib/database/data_structures.h, 92
 lib/database/database.h, 93
 lib/database/dbconn.cpp, 95
 lib/database/dbconn.h, 96
 lib/database/dbconnimp.h, 97
 lib/database/table.cpp, 99
 lib/database/table.h, 100
 lib/database/tablefield.cpp, 101
 lib/database/tablefield.h, 102
 lib/database/ablerow.cpp, 103
 lib/database/ablerow.h, 104
 lib/database_imp/database_imp.h, 106
 lib/database_imp/dummy/dbconn_dummy_imp.cpp, 108
 lib/database_imp/dummy/dbconn_dummy_imp.h, 109
 lib/database_imp/mysql/dbconn_mysql_functions.cpp, 111
 lib/database_imp/mysql/dbconn_mysql_imp.cpp, 112
 lib/database_imp/mysql/dbconn_mysql_imp.h, 113
 lib/database_imp/mysql/dbconn_mysql_result.cpp, 115
 lib/database_imp/mysql/dbconn_mysql_result.h, 116
 lib/dbsql/dbsql.h, 117
 lib/dbsql/dbsql_dummy.h, 118
 lib/dbsql/dbsql_implementations.h, 120
 lib/dbsql/dbsql_mysql.h, 121
 lib/dbsql/dbsqlstatements.cpp, 122
 lib/dbsql/dbsqlstatements.h, 123
 lib/gldb/gldatabase.cpp, 124
 lib/gldb/gldatabase.h, 125
 lib/gldb/gldb.h, 127
 lib/gldb/glexception.h, 128
 lib/gldb/gluser.cpp, 130
 lib/gldb/gluser.h, 130
 lib/gldb/gluser_pass.cpp, 131
 lib/stringhelp/stringhelp.cpp, 133
 lib/stringhelp/stringhelp.h, 133
 load_sample_data
 genleg::GLDatabase, 56
 login
 Database program., 26
 gl_user_main.cpp, 140
 Reporting program., 23

 m_conn
 gldb::DBConnMySQL, 47
 m_data
 gldb::TableField, 78
 m_dbc
 genleg::GLDatabase, 56
 m_enabled
 genleg::GLUser, 62
 m_fields
 gldb::TableRow, 87
 m_firstname
 genleg::GLUser, 62
 m_headers
 gldb::Table, 69
 m_id
 genleg::GLUser, 62
 m_imp
 gldb::DBConn, 37
 m_lastname
 genleg::GLUser, 62
 m_num_fields
 gldb::MySQLResult, 64
 m_opts_set
 genleg::Config, 29
 m_opts_supp
 genleg::Config, 29
 m_pass_hash
 genleg::GLUser, 62
 m_pass_salt
 genleg::GLUser, 62
 m_perms
 genleg::GLUser, 62
 m_quoted
 gldb::Table, 69
 m_records
 gldb::Table, 70
 m_result
 gldb::MySQLResult, 64
 m_sql
 genleg::GLDatabase, 57
 m_tables
 genleg::GLDatabase, 57
 m_username
 genleg::GLUser, 62
 m_views
 genleg::GLDatabase, 57
 main
 Database program., 26
 gl_user_main.cpp, 141
 Reporting program., 23
 MySQLResult
 gldb::MySQLResult, 63

 num_fields
 gldb::MySQLResult, 64
 gldb::Table, 68
 num_records
 gldb::Table, 68

 operator std::string
 gldb::TableField, 76
 operator<<
 gldb::TableField, 78
 operator+=
 gldb::TableField, 76
 operator=

- gldb::DBConn, [36](#)
 - gldb::DBConnDummy, [41](#)
 - gldb::DBConnMySQL, [46](#)
 - gldb::MySQLResult, [64](#)
 - gldb::Table, [68](#), [69](#)
 - gldb::TableField, [76](#), [77](#)
 - gldb::TableRow, [85](#)
- pass_hash
 - genleg::GLUser, [60](#)
- pass_salt
 - genleg::GLUser, [60](#)
- permissions
 - genleg::GLUser, [61](#)
- populate_from_cmdline
 - genleg::Config, [28](#)
- populate_from_file
 - genleg::Config, [29](#)
- print
 - gldb::TableRow, [86](#)
- Program configuration module, [20](#)
- progs/gl_db/gl_db_main.cpp, [135](#)
- progs/gl_report/gl_report_main.cpp, [136](#)
- progs/gl_user/gl_user_main.cpp, [138](#)
- query
 - gldb::DBConn, [36](#)
 - gldb::DBConnDummy, [41](#)
 - gldb::DBConnImp, [44](#)
 - gldb::DBConnMySQL, [46](#)
- record_string
 - gldb::TableRow, [86](#)
- Reporting program., [23](#)
 - login, [23](#)
 - main, [23](#)
 - set_configuration, [24](#)
- result
 - gldb::MySQLResult, [64](#)
- revoke
 - genleg::DBSQLStatements, [51](#)
 - genleg::GLDatabase, [56](#)
- SQL statements module, [19](#)
- select
 - gldb::DBConn, [37](#)
 - gldb::DBConnDummy, [41](#)
 - gldb::DBConnImp, [44](#)
 - gldb::DBConnMySQL, [46](#)
- set_configuration
 - Database program., [26](#)
 - gl_user_main.cpp, [141](#)
 - Reporting program., [24](#)
- set_enabled
 - genleg::GLUser, [61](#)
- set_firstname
 - genleg::GLUser, [61](#)
- set_lastname
 - genleg::GLUser, [61](#)
- set_password
 - genleg::GLUser, [61](#)
- set_quoted
 - gldb::Table, [69](#)
- set_user_password
 - gl_user_main.cpp, [141](#)
- set_username
 - genleg::GLUser, [61](#)
- show_user_details
 - gl_user_main.cpp, [141](#)
- size
 - gldb::TableRow, [86](#)
- split
 - General purpose helpers., [21](#)
- Table
 - gldb::Table, [66](#)
- TableBadInputFile
 - gldb::TableBadInputFile, [71](#)
- TableCouldNotOpenInputFile
 - gldb::TableCouldNotOpenInputFile, [72](#)
- TableException
 - gldb::TableException, [73](#)
- TableField
 - gldb::TableField, [75](#)
- TableMismatchedRecordLength
 - gldb::TableMismatchedRecordLength, [79](#)
- TableNoSuchField
 - gldb::TableNoSuchField, [81](#)
- TableNoSuchRecord
 - gldb::TableNoSuchRecord, [82](#)
- TableRow
 - gldb::TableRow, [83](#), [84](#)
- trim
 - General purpose helpers., [21](#)
- trim_back
 - General purpose helpers., [22](#)
- trim_front
 - General purpose helpers., [22](#)
- update_user
 - genleg::DBSQLStatements, [52](#)
 - genleg::GLDatabase, [56](#)
- user_by_id
 - genleg::DBSQLStatements, [52](#)
- user_by_username
 - genleg::DBSQLStatements, [52](#)
- username
 - genleg::GLUser, [62](#)