

general_ledger

Generated by Doxygen 1.8.1.2

Sun Jun 22 2014 01:49:57

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	Program configuration 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	General Ledger database module.	20
8.4.1	Detailed Description	20
8.4.2	Function Documentation	20
8.4.2.1	decorated_report_from_table	20

8.4.2.2	decorated_row	21
8.4.2.3	grow_widths	21
8.4.2.4	max_column_widths	21
8.4.2.5	plain_report_from_table	21
8.4.2.6	plain_row	22
8.4.2.7	separator_row	22
8.5	General purpose utilities.	23
8.5.1	Detailed Description	24
8.5.2	Function Documentation	24
8.5.2.1	content_lines	24
8.5.2.2	currency_from_string	24
8.5.2.3	join	24
8.5.2.4	next_content_line	24
8.5.2.5	operator!=	25
8.5.2.6	operator+	25
8.5.2.7	operator-	25
8.5.2.8	operator<	25
8.5.2.9	operator<=	26
8.5.2.10	operator==	26
8.5.2.11	operator>	26
8.5.2.12	operator>=	27
8.5.2.13	replace	27
8.5.2.14	split	27
8.5.2.15	split	27
8.5.2.16	split_lines	28
8.5.2.17	trim	28
8.5.2.18	trim_back	28
8.5.2.19	trim_front	28
8.6	Database program.	30
8.6.1	Detailed Description	30
8.6.2	Function Documentation	30
8.6.2.1	check_db_parameters	30
8.6.2.2	check_help_and_version	30
8.6.2.3	login	31
8.6.2.4	main	31
8.6.2.5	set_configuration	31
8.7	Reporting program.	32
8.7.1	Detailed Description	32
8.7.2	Function Documentation	32
8.7.2.1	check_db_parameters	32

8.7.2.2	check_help_and_version	32
8.7.2.3	login	33
8.7.2.4	main	33
8.7.2.5	set_configuration	33
8.8	User administration program.	34
8.8.1	Detailed Description	34
8.8.2	Function Documentation	34
8.8.2.1	check_db_parameters	34
8.8.2.2	check_help_and_version	35
8.8.2.3	check_user_password	35
8.8.2.4	enable_user	35
8.8.2.5	get_user	35
8.8.2.6	login	35
8.8.2.7	main	36
8.8.2.8	set_configuration	36
8.8.2.9	set_user_password	36
8.8.2.10	show_user_details	36
9	Class Documentation	37
9.1	genleg::Config Class Reference	37
9.1.1	Detailed Description	37
9.1.2	Constructor & Destructor Documentation	37
9.1.2.1	Config	37
9.1.2.2	~Config	38
9.1.3	Member Function Documentation	38
9.1.3.1	add_cmdline_option	38
9.1.3.2	is_set	38
9.1.3.3	operator[]	38
9.1.3.4	populate_from_cmdline	38
9.1.3.5	populate_from_file	39
9.1.4	Member Data Documentation	39
9.1.4.1	m_opts_set	39
9.1.4.2	m_opts_supp	39
9.2	genleg::ConfigBadConfigFile Class Reference	39
9.2.1	Detailed Description	40
9.2.2	Constructor & Destructor Documentation	40
9.2.2.1	ConfigBadConfigFile	40
9.3	genleg::ConfigBadOption Class Reference	41
9.3.1	Detailed Description	41
9.3.2	Constructor & Destructor Documentation	41

9.3.2.1	ConfigBadOption	42
9.4	genleg::ConfigCouldNotOpenFile Class Reference	42
9.4.1	Detailed Description	43
9.4.2	Constructor & Destructor Documentation	43
9.4.2.1	ConfigCouldNotOpenFile	43
9.5	genleg::ConfigException Class Reference	43
9.5.1	Detailed Description	43
9.5.2	Constructor & Destructor Documentation	44
9.5.2.1	ConfigException	44
9.6	genleg::ConfigOptionNotSet Class Reference	44
9.6.1	Detailed Description	45
9.6.2	Constructor & Destructor Documentation	45
9.6.2.1	ConfigOptionNotSet	45
9.7	pgutils::Currency Class Reference	45
9.7.1	Detailed Description	46
9.7.2	Constructor & Destructor Documentation	46
9.7.2.1	Currency	46
9.7.3	Member Function Documentation	46
9.7.3.1	expand	46
9.7.3.2	operator+=	46
9.7.3.3	operator-	46
9.7.3.4	operator-=	47
9.7.4	Friends And Related Function Documentation	47
9.7.4.1	operator+	47
9.7.4.2	operator<	47
9.7.4.3	operator==	47
9.7.5	Member Data Documentation	48
9.7.5.1	m_frac	48
9.7.5.2	m_int	48
9.8	pgutils::CurrencyException Class Reference	48
9.8.1	Detailed Description	48
9.8.2	Constructor & Destructor Documentation	48
9.8.2.1	CurrencyException	48
9.9	gldb::DBConn Class Reference	48
9.9.1	Detailed Description	49
9.9.2	Constructor & Destructor Documentation	49
9.9.2.1	DBConn	49
9.9.2.2	DBConn	50
9.9.2.3	DBConn	50
9.9.3	Member Function Documentation	50

9.9.3.1	last_auto_increment	50
9.9.3.2	operator=	50
9.9.3.3	operator=	50
9.9.3.4	query	50
9.9.3.5	select	50
9.9.4	Member Data Documentation	50
9.9.4.1	m_imp	51
9.10	gldb::DBConnCouldNotConnect Class Reference	51
9.10.1	Detailed Description	52
9.10.2	Constructor & Destructor Documentation	52
9.10.2.1	DBConnCouldNotConnect	52
9.11	gldb::DBConnCouldNotQuery Class Reference	52
9.11.1	Detailed Description	53
9.11.2	Constructor & Destructor Documentation	53
9.11.2.1	DBConnCouldNotQuery	53
9.12	gldb::DBConnDummy Class Reference	53
9.12.1	Detailed Description	54
9.12.2	Constructor & Destructor Documentation	54
9.12.2.1	DBConnDummy	55
9.12.2.2	DBConnDummy	55
9.12.2.3	~DBConnDummy	55
9.12.3	Member Function Documentation	55
9.12.3.1	operator=	55
9.12.3.2	query	55
9.12.3.3	select	55
9.13	gldb::DBConnException Class Reference	56
9.13.1	Detailed Description	56
9.13.2	Constructor & Destructor Documentation	56
9.13.2.1	DBConnException	56
9.14	gldb::DBConnImp Class Reference	56
9.14.1	Detailed Description	57
9.14.2	Constructor & Destructor Documentation	57
9.14.2.1	DBConnImp	57
9.14.2.2	~DBConnImp	57
9.14.3	Member Function Documentation	57
9.14.3.1	last_auto_increment	57
9.14.3.2	query	58
9.14.3.3	select	58
9.15	gldb::DBConnMySQL Class Reference	58
9.15.1	Detailed Description	59

9.15.2	Constructor & Destructor Documentation	59
9.15.2.1	DBConnMySQL	59
9.15.2.2	DBConnMySQL	60
9.15.2.3	DBConnMySQL	60
9.15.2.4	~DBConnMySQL	60
9.15.3	Member Function Documentation	60
9.15.3.1	last_auto_increment	60
9.15.3.2	operator=	60
9.15.3.3	operator=	60
9.15.3.4	query	60
9.15.3.5	select	61
9.15.4	Member Data Documentation	61
9.15.4.1	m_conn	61
9.15.4.2	m_mtx	61
9.16	genleg::DBSQLDummy Class Reference	61
9.16.1	Detailed Description	62
9.17	genleg::DBSQLMySQL Class Reference	62
9.17.1	Detailed Description	63
9.18	genleg::DBSQLStatements Class Reference	63
9.18.1	Detailed Description	65
9.18.2	Constructor & Destructor Documentation	65
9.18.2.1	DBSQLStatements	65
9.18.2.2	~DBSQLStatements	65
9.18.3	Member Function Documentation	65
9.18.3.1	create_table	65
9.18.3.2	create_view	65
9.18.3.3	currenttb	65
9.18.3.4	currenttb_by_entity	65
9.18.3.5	drop_table	66
9.18.3.6	drop_view	66
9.18.3.7	get_perms	66
9.18.3.8	grant	66
9.18.3.9	listusers	67
9.18.3.10	post_je	67
9.18.3.11	post_je_line	67
9.18.3.12	revoke	67
9.18.3.13	update_user	68
9.18.3.14	user_by_id	68
9.18.3.15	user_by_username	68
9.19	genleg::GLDatabase Class Reference	68

9.19.1 Detailed Description	70
9.19.2 Constructor & Destructor Documentation	70
9.19.2.1 GLDatabase	70
9.19.2.2 ~GLDatabase	70
9.19.3 Member Function Documentation	70
9.19.3.1 backend	71
9.19.3.2 create_structure	71
9.19.3.3 create_user	71
9.19.3.4 current_trial_balance_report	71
9.19.3.5 destroy_structure	71
9.19.3.6 get_user_by_id	72
9.19.3.7 get_user_by_username	72
9.19.3.8 grant	72
9.19.3.9 list_users_report	72
9.19.3.10 load_sample_data	72
9.19.3.11 post_journal	73
9.19.3.12 report	73
9.19.3.13 revoke	73
9.19.3.14 update_user	73
9.19.4 Member Data Documentation	73
9.19.4.1 m_dbc	73
9.19.4.2 m_sql	74
9.19.4.3 m_tables	74
9.19.4.4 m_views	74
9.20 genleg::GLDBException Class Reference	74
9.20.1 Detailed Description	74
9.20.2 Constructor & Destructor Documentation	74
9.20.2.1 GLDBException	74
9.21 genleg::GLJELine Class Reference	74
9.21.1 Detailed Description	75
9.21.2 Member Function Documentation	75
9.21.2.1 account	75
9.21.2.2 amount	75
9.21.3 Member Data Documentation	76
9.21.3.1 m_acct	76
9.21.3.2 m_amount	76
9.22 genleg::GLJournal Class Reference	76
9.22.1 Detailed Description	77
9.22.2 Constructor & Destructor Documentation	77
9.22.2.1 GLJournal	77

9.22.3	Member Function Documentation	77
9.22.3.1	begin	77
9.22.3.2	begin	77
9.22.3.3	end	77
9.22.3.4	end	78
9.22.4	Member Data Documentation	78
9.22.4.1	m_entity	78
9.22.4.2	m_lines	78
9.22.4.3	m_memo	78
9.22.4.4	m_period	78
9.22.4.5	m_source	78
9.22.4.6	m_year	78
9.23	genleg::GLReport Class Reference	78
9.23.1	Detailed Description	79
9.23.2	Constructor & Destructor Documentation	79
9.23.2.1	GLReport	79
9.23.2.2	~GLReport	79
9.23.3	Friends And Related Function Documentation	79
9.23.3.1	operator<<	79
9.23.4	Member Data Documentation	79
9.23.4.1	m_report_text	79
9.24	genleg::GLUser Class Reference	79
9.24.1	Detailed Description	81
9.24.2	Constructor & Destructor Documentation	81
9.24.2.1	GLUser	81
9.24.2.2	~GLUser	81
9.24.3	Member Function Documentation	81
9.24.3.1	check_password	81
9.24.3.2	enabled	82
9.24.3.3	firstname	82
9.24.3.4	id	82
9.24.3.5	lastname	82
9.24.3.6	pass_hash	82
9.24.3.7	pass_salt	82
9.24.3.8	permissions	83
9.24.3.9	set_enabled	83
9.24.3.10	set_firstname	83
9.24.3.11	set_lastname	83
9.24.3.12	set_password	83
9.24.3.13	set_username	83

9.24.3.14 username	83
9.24.4 Member Data Documentation	84
9.24.4.1 m_enabled	84
9.24.4.2 m_firstname	84
9.24.4.3 m_id	84
9.24.4.4 m_lastname	84
9.24.4.5 m_pass_hash	84
9.24.4.6 m_pass_salt	84
9.24.4.7 m_perms	84
9.24.4.8 m_username	84
9.25 glldb::MySQLResult Class Reference	84
9.25.1 Detailed Description	85
9.25.2 Constructor & Destructor Documentation	85
9.25.2.1 MySQLResult	85
9.25.2.2 ~MySQLResult	85
9.25.2.3 MySQLResult	85
9.25.2.4 MySQLResult	85
9.25.3 Member Function Documentation	85
9.25.3.1 num_fields	86
9.25.3.2 operator=	86
9.25.3.3 operator=	86
9.25.3.4 result	86
9.25.4 Member Data Documentation	86
9.25.4.1 m_num_fields	86
9.25.4.2 m_result	86
9.26 glldb::Table Class Reference	86
9.26.1 Detailed Description	88
9.26.2 Constructor & Destructor Documentation	88
9.26.2.1 Table	88
9.26.2.2 Table	88
9.26.2.3 Table	88
9.26.2.4 Table	89
9.26.2.5 ~Table	89
9.26.3 Member Function Documentation	89
9.26.3.1 append_record	89
9.26.3.2 append_record	89
9.26.3.3 begin	89
9.26.3.4 begin	89
9.26.3.5 create_from_file	90
9.26.3.6 end	90

9.26.3.7	end	90
9.26.3.8	get_field	90
9.26.3.9	get_headers	91
9.26.3.10	insert_query	91
9.26.3.11	num_fields	91
9.26.3.12	num_records	91
9.26.3.13	operator=	91
9.26.3.14	operator=	91
9.26.3.15	operator[]	92
9.26.3.16	set_quoted	92
9.26.3.17	set_quoted	92
9.26.4	Member Data Documentation	92
9.26.4.1	m_headers	92
9.26.4.2	m_quoted	92
9.26.4.3	m_records	92
9.27	gldb::TableBadInputFile Class Reference	93
9.27.1	Detailed Description	93
9.27.2	Constructor & Destructor Documentation	93
9.27.2.1	TableBadInputFile	94
9.28	gldb::TableCouldNotOpenInputFile Class Reference	94
9.28.1	Detailed Description	95
9.28.2	Constructor & Destructor Documentation	95
9.28.2.1	TableCouldNotOpenInputFile	95
9.29	gldb::TableException Class Reference	95
9.29.1	Detailed Description	96
9.29.2	Constructor & Destructor Documentation	96
9.29.2.1	TableException	96
9.30	gldb::TableField Class Reference	96
9.30.1	Detailed Description	97
9.30.2	Constructor & Destructor Documentation	97
9.30.2.1	TableField	97
9.30.2.2	TableField	97
9.30.2.3	TableField	98
9.30.2.4	TableField	98
9.30.2.5	TableField	98
9.30.2.6	~TableField	98
9.30.3	Member Function Documentation	98
9.30.3.1	length	98
9.30.3.2	operator std::string	98
9.30.3.3	operator+=	98

9.30.3.4	operator+=	99
9.30.3.5	operator=	99
9.30.3.6	operator=	99
9.30.3.7	operator=	99
9.30.3.8	operator=	100
9.30.3.9	operator=	100
9.30.3.10	operator[]	100
9.30.3.11	operator[]	100
9.30.4	Friends And Related Function Documentation	101
9.30.4.1	operator<<	101
9.30.5	Member Data Documentation	101
9.30.5.1	m_data	101
9.31	gldb::TableMismatchedRecordLength Class Reference	101
9.31.1	Detailed Description	102
9.31.2	Constructor & Destructor Documentation	102
9.31.2.1	TableMismatchedRecordLength	102
9.32	gldb::TableNoSuchField Class Reference	102
9.32.1	Detailed Description	103
9.32.2	Constructor & Destructor Documentation	103
9.32.2.1	TableNoSuchField	103
9.33	gldb::TableNoSuchRecord Class Reference	104
9.33.1	Detailed Description	104
9.33.2	Constructor & Destructor Documentation	104
9.33.2.1	TableNoSuchRecord	105
9.34	gldb::TableRow Class Reference	105
9.34.1	Detailed Description	106
9.34.2	Constructor & Destructor Documentation	106
9.34.2.1	TableRow	106
9.34.2.2	TableRow	106
9.34.2.3	TableRow	106
9.34.2.4	TableRow	106
9.34.2.5	TableRow	107
9.34.2.6	TableRow	107
9.34.2.7	TableRow	107
9.34.2.8	~TableRow	107
9.34.3	Member Function Documentation	107
9.34.3.1	append_field	107
9.34.3.2	append_field	107
9.34.3.3	append_field	108
9.34.3.4	append_field	108

9.34.3.5	append_field	108
9.34.3.6	begin	108
9.34.3.7	begin	108
9.34.3.8	end	108
9.34.3.9	end	108
9.34.3.10	operator=	109
9.34.3.11	operator=	109
9.34.3.12	operator[]	109
9.34.3.13	operator[]	109
9.34.3.14	print	110
9.34.3.15	record_string	110
9.34.3.16	record_string	110
9.34.3.17	size	110
9.34.4	Member Data Documentation	110
9.34.4.1	m_fields	110
10	File Documentation	111
10.1	lib/config/config.cpp File Reference	111
10.1.1	Detailed Description	111
10.2	lib/config/config.h File Reference	112
10.2.1	Detailed Description	113
10.3	lib/config/config_getopt.cpp File Reference	113
10.3.1	Detailed Description	113
10.3.2	Macro Definition Documentation	114
10.3.2.1	_XOPEN_SOURCE	114
10.4	lib/database/data_structures.h File Reference	114
10.4.1	Detailed Description	115
10.5	lib/database/database.h File Reference	115
10.5.1	Detailed Description	117
10.6	lib/database/dbconn.cpp File Reference	117
10.6.1	Detailed Description	117
10.7	lib/database/dbconn.h File Reference	118
10.7.1	Detailed Description	119
10.8	lib/database/dbconnimp.h File Reference	119
10.8.1	Detailed Description	121
10.9	lib/database/table.cpp File Reference	121
10.9.1	Detailed Description	121
10.10	lib/database/table.h File Reference	122
10.10.1	Detailed Description	123
10.11	lib/database/tablefield.cpp File Reference	124

10.11.1 Detailed Description	124
10.12lib/database/tablefield.h File Reference	124
10.12.1 Detailed Description	126
10.13lib/database/tablerow.cpp File Reference	126
10.13.1 Detailed Description	126
10.14lib/database/tablerow.h File Reference	127
10.14.1 Detailed Description	128
10.15lib/database_imp/database_imp.h File Reference	128
10.15.1 Detailed Description	130
10.16lib/database_imp/dummy/dbconn_dummy_imp.cpp File Reference	130
10.16.1 Detailed Description	131
10.17lib/database_imp/dummy/dbconn_dummy_imp.h File Reference	131
10.17.1 Detailed Description	133
10.18lib/database_imp/mysql/dbconn_mysql_functions.cpp File Reference	133
10.18.1 Detailed Description	134
10.19lib/database_imp/mysql/dbconn_mysql_imp.cpp File Reference	134
10.19.1 Detailed Description	135
10.20lib/database_imp/mysql/dbconn_mysql_imp.h File Reference	136
10.20.1 Detailed Description	137
10.21lib/database_imp/mysql/dbconn_mysql_result.cpp File Reference	137
10.21.1 Detailed Description	138
10.22lib/database_imp/mysql/dbconn_mysql_result.h File Reference	138
10.22.1 Detailed Description	139
10.23lib/dbsql/dbsql.h File Reference	139
10.23.1 Detailed Description	140
10.24lib/dbsql/dbsql_dummy.h File Reference	140
10.24.1 Detailed Description	141
10.25lib/dbsql/dbsql_functions.h File Reference	142
10.25.1 Detailed Description	143
10.26lib/dbsql/dbsql_implementations.h File Reference	143
10.26.1 Detailed Description	144
10.27lib/dbsql/dbsql_mysql.h File Reference	145
10.27.1 Detailed Description	146
10.28lib/dbsql/dbsqlstatements.cpp File Reference	146
10.28.1 Detailed Description	146
10.29lib/dbsql/dbsqlstatements.h File Reference	147
10.29.1 Detailed Description	148
10.30lib/gldb/gldatabase.cpp File Reference	148
10.30.1 Detailed Description	149
10.30.2 Function Documentation	149

10.30.2.1 boolstring_to_bool	149
10.31lib/gldb/gldatabase.h File Reference	149
10.31.1 Detailed Description	150
10.32lib/gldb/gldb.h File Reference	151
10.32.1 Detailed Description	151
10.33lib/gldb/glexception.h File Reference	152
10.33.1 Detailed Description	153
10.34lib/gldb/gljournal.cpp File Reference	153
10.34.1 Detailed Description	153
10.35lib/gldb/gljournal.h File Reference	154
10.35.1 Detailed Description	155
10.36lib/gldb/glreport.cpp File Reference	155
10.36.1 Detailed Description	156
10.37lib/gldb/glreport.h File Reference	157
10.37.1 Detailed Description	158
10.38lib/gldb/gluser.cpp File Reference	158
10.38.1 Detailed Description	159
10.39lib/gldb/gluser.h File Reference	159
10.39.1 Detailed Description	160
10.40lib/gldb/gluser_pass.cpp File Reference	160
10.40.1 Detailed Description	161
10.40.2 Macro Definition Documentation	161
10.40.2.1 _XOPEN_SOURCE	161
10.40.3 Function Documentation	162
10.40.3.1 generate_salt	162
10.41lib/pgutils/currency.cpp File Reference	162
10.41.1 Detailed Description	162
10.42lib/pgutils/currency.h File Reference	162
10.42.1 Detailed Description	164
10.43lib/pgutils/pgutils.h File Reference	164
10.43.1 Detailed Description	165
10.44lib/pgutils/stringhelp.cpp File Reference	165
10.44.1 Detailed Description	166
10.45lib/pgutils/stringhelp.h File Reference	166
10.45.1 Detailed Description	167
10.46progs/gl_db/gl_db_main.cpp File Reference	167
10.46.1 Detailed Description	168
10.47progs/gl_report/gl_report_main.cpp File Reference	169
10.47.1 Detailed Description	170
10.48progs/gl_user/gl_user_main.cpp File Reference	170

10.48.1 Detailed Description	171
--	-----

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:

Program configuration module	15
Database interaction module	16
SQL statements module	19
General Ledger database module.	20
General purpose utilities.	23
Database program.	30
Reporting program.	32
User administration program.	34

Chapter 5

Class Index

5.1 Class Hierarchy

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

genleg::Config	37
genleg::ConfigException	43
genleg::ConfigBadConfigFile	39
genleg::ConfigBadOption	41
genleg::ConfigCouldNotOpenFile	42
genleg::ConfigOptionNotSet	44
pgutils::Currency	45
pgutils::CurrencyException	48
gldb::DBConn	48
gldb::DBConnException	56
gldb::DBConnCouldNotConnect	51
gldb::DBConnCouldNotQuery	52
gldb::DBConnImp	56
gldb::DBConnDummy	53
gldb::DBConnMySQL	58
genleg::DBSQLStatements	63
genleg::DBSQLDummy	61
genleg::DBSQLMySQL	62
genleg::GLDatabase	68
genleg::GLDBException	74
genleg::GLJELine	74
genleg::GLJournal	76
genleg::GLReport	78
genleg::GLUser	79
gldb::MySQLResult	84
gldb::Table	86
gldb::TableException	95
gldb::TableBadInputFile	93
gldb::TableCouldNotOpenInputFile	94
gldb::TableMismatchedRecordLength	101
gldb::TableNoSuchField	102
gldb::TableNoSuchRecord	104
gldb::TableField	96
gldb::TableRow	105

Chapter 6

Class Index

6.1 Class List

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

genleg::Config	Configuration options class	37
genleg::ConfigBadConfigFile	Exception class for badly formed configuration file	39
genleg::ConfigBadOption	Exception class for bad provided option	41
genleg::ConfigCouldNotOpenFile	Exception class for when conf file cannot be opened	42
genleg::ConfigException	Configuration module exception base class	43
genleg::ConfigOptionNotSet	Exception class for option not set	44
pgutils::Currency	Currency amount class	45
pgutils::CurrencyException	Base Currency exception class	48
gldb::DBConn	Database connection class	48
gldb::DBConnCouldNotConnect	Could not connect to database exception class	51
gldb::DBConnCouldNotQuery	Could not execute database query exception class	52
gldb::DBConnDummy	Dummy database implementation class	53
gldb::DBConnException	Base database connection exception class	56
gldb::DBConnImp	Abstract database implementation base class	56
gldb::DBConnMySQL	MySQL database implementation class	58
genleg::DBSQLDummy	Dummy SQL statements class	61
genleg::DBSQLMySQL	MySQL SQL statements class	62
genleg::DBSQLStatements	SQL statements class	63
genleg::GLDatabase	General ledger database class	68

genleg::GLDBException	
Base general ledger database exceptionc class	74
genleg::GLJELine	
Journal entry line class	74
genleg::GLJournal	
Journal entry class	76
genleg::GLReport	
General ledger report class	78
genleg::GLUser	
General ledger user class	79
gldb::MySQLResult	
MySQL result structure class	84
gldb::Table	
Database table class	86
gldb::TableBadInputFile	
Could not connect to database exception class	93
gldb::TableCouldNotOpenInputFile	
Could not connect to database exception class	94
gldb::TableException	
Base database connection exception class	95
gldb::TableField	
Database table field class	96
gldb::TableMismatchedRecordLength	
Mismatched record length exception class	101
gldb::TableNoSuchField	
No such field exception class	102
gldb::TableNoSuchRecord	
No such record exception class	104
gldb::TableRow	
Database table row class	105

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	111
lib/config/ config.h	
Interface to program configurations class	112
lib/config/ config_getopt.cpp	
Implementation of command line functionality	113
lib/database/ data_structures.h	
Main interface to database data structures	114
lib/database/ database.h	
User interface to database functionality	115
lib/database/ dbconn.cpp	
Implementation of database connection class	117
lib/database/ dbconn.h	
Interface to database connection base class	118
lib/database/ dbconnimp.h	
Interface to abstract database implementation base class	119
lib/database/ table.cpp	
Implementation of database table data structure	121
lib/database/ table.h	
Interface to database table data structure	122
lib/database/ tablefield.cpp	
Implementation of database table field class	124
lib/database/ tablefield.h	
Interface to database table field class	124
lib/database/ tablerow.cpp	
Implementation of database table row data structure	126
lib/database/ tablerow.h	
Interface to database table row data structure	127
lib/database_imp/ database_imp.h	
Interface to database implementation factory function	128
lib/database_imp/dummy/ dbconn_dummy_imp.cpp	
Implementation of Dummy database connection implementation class	130
lib/database_imp/dummy/ dbconn_dummy_imp.h	
Interface to dummy database connection implementation class	131
lib/database_imp/mysql/ dbconn_mysql_functions.cpp	
Implementation of MySQL implementation factory function	133
lib/database_imp/mysql/ dbconn_mysql_imp.cpp	
Implementation of MySQL database connection implementation class	134

lib/database_imp/mysql/dbconn_mysql_imp.h	
Interface to MySQL database connection implementation class	136
lib/database_imp/mysql/dbconn_mysql_result.cpp	
Implementation of MySQL result structure resource handle class	137
lib/database_imp/mysql/dbconn_mysql_result.h	
Interface to MySQL result structure resource handle class	138
lib/dbsql/dbsql.h	
User interface to DBSQL module	139
lib/dbsql/dbsql_dummy.h	
Interface to dummy SQL statement class	140
lib/dbsql/dbsql_functions.h	
Interface to SQL module standalone functions	142
lib/dbsql/dbsql_implementations.h	
Aggregation header for DBSQLStatements implementations	143
lib/dbsql/dbsql_mysql.h	
Interface to MySQL SQL statement class	145
lib/dbsql/dbsqlstatements.cpp	
Implementation of SQL statement class	146
lib/dbsql/dbsqlstatements.h	
Implementation of SQL module standalone functions	147
lib/gldb/gldatabase.cpp	
Implementation of General Ledger database class	148
lib/gldb/gldatabase.h	
Interface to General Ledger database class	149
lib/gldb/gldb.h	
User interface to General Ledger database module	151
lib/gldb/glexception.h	
Interface to General Ledger base exception class	152
lib/gldb/gljournal.cpp	
Implementation of journal entry classes	153
lib/gldb/gljournal.h	
Interface to journal entry classes	154
lib/gldb/glreport.cpp	
Implementation of report class	155
lib/gldb/glreport.h	
Interface to report class	157
lib/gldb/gluser.cpp	
Implementation of user class	158
lib/gldb/gluser.h	
Interface to user class	159
lib/gldb/gluser_pass.cpp	
Implementation of password functions for user class	160
lib/pgutils/currency.cpp	
Implementation of currency amount class	162
lib/pgutils/currency.h	
Interface to currency amount class	162
lib/pgutils/pgutils.h	
Aggregate interface to general utility functions	164
lib/pgutils/stringhelp.cpp	
Implementation of string helper functions	165
lib/pgutils/stringhelp.h	
Interface to string helper functions	166
progs/gl_db/gl_db_main.cpp	
Main functionality for gl_db program	167
progs/gl_report/gl_report_main.cpp	
Main functionality for gl_report program	169
progs/gl_user/gl_user_main.cpp	
Main functionality for gl_user program	170

Chapter 8

Module Documentation

8.1 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.

8.1.1 Detailed Description

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

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 General Ledger database module.

Classes

- class `genleg::GLDatabase`
General ledger database class.
- class `genleg::GLDBException`
Base general ledger database exception class.
- class `genleg::GLJELine`
Journal entry line class.
- class `genleg::GLJournal`
Journal entry class.
- class `genleg::GLReport`
General ledger report class.
- class `genleg::GLUser`
General ledger user class.
- class `pgutils::CurrencyException`
*Base *Currency* exception class.*

Functions

- static `std::vector< size_t > max_column_widths` (const `gldb::Table` &table)
Calculates the maximum required column widths for a table.
- static void `grow_widths` (std::vector< size_t > &widths, const `TableRow` &row)
Increments a vector of required column widths.
- static `std::string separator_row` (const std::vector< size_t > &widths)
Returns a decorated separator row for a table.
- static `std::string plain_row` (const `TableRow` &row, const std::vector< size_t > &widths)
Returns a row for a plain report.
- static `std::string decorated_row` (const `TableRow` &row, const std::vector< size_t > &widths)
Returns a row for a decorated report.
- `std::string genleg::plain_report_from_table` (const `gldb::Table` &table)
Creates a plain report from a table.
- `std::string genleg::decorated_report_from_table` (const `gldb::Table` &table)
Creates a decorated report from a table.

8.4.1 Detailed Description

Module for interacting with the general ledger database model.

8.4.2 Function Documentation

8.4.2.1 `std::string genleg::decorated_report_from_table (const gldb::Table & table)`

Creates a decorated report from a table.

A "decorated report" presents the table surrounding with ASCII-art style lines consisting of '+' , '-' and '|' characters.

Parameters

<i>table</i>	The table from which to create the report.
--------------	--

Returns

A string containing the report.

8.4.2.2 `static std::string decorated_row (const TableRow & row, const std::vector< size_t > & widths) [static]`

Returns a row for a decorated report.

Parameters

<i>row</i>	The row for which to create the report row.
<i>widths</i>	A vector of required widths.

Returns

A string containing the decorated row.

8.4.2.3 `static void grow_widths (std::vector< size_t > & widths, const TableRow & row) [static]`

Increments a vector of required column widths.

Each element of the vector is increased to fit the width of each file in the row, if the existing width is not large enough to contain it.

Parameters

<i>widths</i>	An existing vector of widths.
<i>row</i>	The row against which to check and potentially increase the vector.

8.4.2.4 `static std::vector< size_t > max_column_widths (const glldb::Table & table) [static]`

Calculates the maximum required column widths for a table.

Parameters

<i>table</i>	The table.
--------------	------------

Returns

A vector of `size_t` containing the maximum required width for each column, without padding.

8.4.2.5 `std::string genleg::plain_report_from_table (const glldb::Table & table)`

Creates a plain report from a table.

A "plain report" separates each column with a space.

Parameters

<i>table</i>	The table from which to create the report.
--------------	--

Returns

A string containing the report.

8.4.2.6 `static std::string plain_row (const TableRow & row, const std::vector< size_t > & widths)` `[static]`

Returns a row for a plain report.

Parameters

<i>row</i>	The row for which to create the report row.
<i>widths</i>	A vector of required widths.

Returns

A string containing the plain row.

8.4.2.7 `static std::string separator_row (const std::vector< size_t > & widths)` `[static]`

Returns a decorated separator row for a table.

The "separator row" is of the format "+—+—+—+" where each column is separated by a ' +' character, and consists of enough ' - ' characters to fit the respective width in the vector plus two additional characters for spacing.

Parameters

<i>widths</i>	A vector of required widths.
---------------	------------------------------

Returns

A string containing the separator row.

8.5 General purpose utilities.

Classes

- class `pgutils::Currency`
Currency amount class.

Functions

- `Currency pgutils::operator+ (Currency lhs, const Currency &rhs)`
Currency addition operator.
- `Currency pgutils::operator- (Currency lhs, const Currency &rhs)`
Currency subtraction operator.
- `bool pgutils::operator== (const Currency &lhs, const Currency &rhs)`
Currency equality comparison operator.
- `bool pgutils::operator!= (const Currency &lhs, const Currency &rhs)`
Currency inequality comparison operator.
- `bool pgutils::operator< (const Currency &lhs, const Currency &rhs)`
Currency less than comparison operator.
- `bool pgutils::operator> (const Currency &lhs, const Currency &rhs)`
Currency greater than comparison operator.
- `bool pgutils::operator<= (const Currency &lhs, const Currency &rhs)`
Currency less than or equal to comparison operator.
- `bool pgutils::operator>= (const Currency &lhs, const Currency &rhs)`
Currency greater than or equal to comparison operator.
- `Currency pgutils::currency_from_string (const std::string &s)`
Creates a currency amount from a string representation.
- `std::string & pgutils::trim_front (std::string &s)`
Trims leading whitespace from a string.
- `std::string & pgutils::trim_back (std::string &s)`
Trims trailing whitespace from a string.
- `std::string & pgutils::trim (std::string &s)`
Trims leading and trailing whitespace from a string.
- `std::vector< std::string > pgutils::split (const std::string &s, const char delim)`
Splits a delimited string into tokens.
- `std::vector< std::string > & pgutils::split (std::vector< std::string > &vec, const std::string &s, const char delim)`
Splits a delimited string into tokens.
- `bool pgutils::next_content_line (std::istream &ifs, std::string &s)`
Gets the next content line from a stream.
- `std::vector< std::string > & pgutils::content_lines (std::vector< std::string > &vec, std::istream &ifs)`
Populates a vector of content lines from a stream.
- `std::vector< std::vector< std::string > > & pgutils::split_lines (std::vector< std::vector< std::string > > &vec, std::istream &ifs, const char delim)`
Populates a vector of vectors of fields from a stream.
- `std::string & pgutils::join (const std::vector< std::string > &vec, std::string &s, const char delim)`
Joins a vector of strings into a delimited line.
- `bool pgutils::replace (std::string &str, const std::string &from, const std::string &to)`
Replaces a substring with another string.

8.5.1 Detailed Description

General purpose utility classes and functions.

8.5.2 Function Documentation

8.5.2.1 `std::vector< std::string > & pgutils::content_lines (std::vector< std::string > & vec, std::istream & ifs)`

Populates a vector of content lines from a stream.

Parameters

<i>vec</i>	The vector to populate.
<i>ifs</i>	The input stream.

Returns

A reference to `vec`.

8.5.2.2 `pgutils::currency_from_string (const std::string & s)`

Creates a currency amount from a string representation.

Parameters

<i>s</i>	The string representation.
----------	----------------------------

Returns

The currency representation.

8.5.2.3 `std::string & pgutils::join (const std::vector< std::string > & vec, std::string & s, const char delim)`

Joins a vector of strings into a delimited line.

The function is the opposite of `split`.

Parameters

<i>vec</i>	The vector containing the strings.
<i>s</i>	The string in which to store the line.
<i>delim</i>	The delimiter character to be used to delimit the fields.

Returns

A reference to `s`.

8.5.2.4 `bool pgutils::next_content_line (std::istream & ifs, std::string & s)`

Gets the next content line from a stream.

A "content line" is defined as a line which, after being trimmed of trailing and leading whitespace, is not empty, and does not start with a '#' character (indicating a comment line).

Parameters

<i>ifs</i>	The input stream.
<i>s</i>	The string in which to store the line.

Returns

`true` if there is a next content line, `false` otherwise.

8.5.2.5 `bool pgutils::operator!=(const Currency & lhs, const Currency & rhs)`

[Currency](#) inequality comparison operator.

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Return values

<i>true</i>	If the two sides are not equal.
<i>false</i>	If the two sides are equal.

8.5.2.6 `Currency pgutils::operator+ (Currency lhs, const Currency & rhs)`

[Currency](#) addition operator.

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Returns

The sum of the two sides.

8.5.2.7 `Currency pgutils::operator- (Currency lhs, const Currency & rhs)`

[Currency](#) subtraction operator.

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Returns

The difference between the two sides.

8.5.2.8 `bool pgutils::operator< (const Currency & lhs, const Currency & rhs)`

[Currency](#) less than comparison operator.

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Return values

<i>true</i>	If the lhs is less than the rhs.
<i>false</i>	If the lhs is not less than the rhs.

8.5.2.9 `bool pgutils::operator<= (const Currency & lhs, const Currency & rhs)`

[Currency](#) less than or equal to comparison operator.

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Return values

<i>true</i>	If the lhs is less than or equal to the rhs.
<i>false</i>	If the lhs is not less than or equal to the rhs.

8.5.2.10 `bool pgutils::operator== (const Currency & lhs, const Currency & rhs)`

[Currency](#) equality comparison operator.

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Return values

<i>true</i>	If the two sides are equal.
<i>false</i>	If the two sides are not equal.

8.5.2.11 `bool pgutils::operator> (const Currency & lhs, const Currency & rhs)`

[Currency](#) greater than comparison operator.

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Return values

<i>true</i>	If the lhs is greater than the rhs.
<i>false</i>	If the lhs is not greater than the rhs.

8.5.2.12 `bool pgutils::operator>= (const Currency & lhs, const Currency & rhs)`

`Currency` greater than or equal to comparison operator.

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Return values

<i>true</i>	If the lhs is greater than or equal to the rhs.
<i>false</i>	If the lhs is not greater than or equal to the rhs.

8.5.2.13 `bool pgutils::replace (std::string & str, const std::string & from, const std::string & to)`

Replaces a substring with another string.

Parameters

<i>str</i>	The string containing the substring to replace.
<i>from</i>	The substring to replace.
<i>to</i>	The string with which to replace the substring.

Returns

`true` if a replacement was made, `false` otherwise.

8.5.2.14 `std::vector< std::string > pgutils::split (const std::string & s, const char delim)`

Splits a delimited string into tokens.

Parameters

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

Returns

A vector of tokens.

8.5.2.15 `std::vector< std::string > & pgutils::split (std::vector< std::string > & vec, const std::string & s, const char delim)`

Splits a delimited string into tokens.

Parameters

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

Returns

A reference to `vec`.

8.5.2.16 `std::vector< std::vector< std::string > > & pgutils::split_lines (std::vector< std::vector< std::string >> & vec, std::istream & ifs, const char delim)`

Populates a vector of vectors of fields from a stream.

Parameters

<i>vec</i>	The vector to populate.
<i>ifs</i>	The input stream.
<i>delim</i>	The delimiter character to split each content line.

Returns

A reference to `vec`.

8.5.2.17 `std::string & pgutils::trim (std::string & s)`

Trims leading and trailing whitespace from a string.

Parameters

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

Returns

The trimmed string.

8.5.2.18 `std::string & pgutils::trim_back (std::string & s)`

Trims trailing whitespace from a string.

Parameters

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

Returns

The trimmed string.

8.5.2.19 `std::string & pgutils::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 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 * `programe` = "gl_db"
Static variable for program name.

8.6.1 Detailed Description

Administrative database management program.

8.6.2 Function Documentation

8.6.2.1 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.

8.6.2.2 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.

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

Gets a password from the terminal.

Returns

The password.

8.6.2.4 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.

8.6.2.5 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> .

8.7 Reporting 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 * `programe` = "gl_report"
Static variable for program name.

8.7.1 Detailed Description

Administrative reporting 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> .

8.8 User administration 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.
- `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 * `progrname` = "gl_user"
Static variable for program name.

8.8.1 Detailed Description

User administration program.

8.8.2 Function Documentation

8.8.2.1 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.

8.8.2.2 `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.

8.8.2.3 `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.

8.8.2.4 `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.

8.8.2.5 `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.

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

Gets a password from the terminal.

Returns

The password.

8.8.2.7 `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.

8.8.2.8 `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 <code>main()</code> .
<i>argv</i>	<i>argv</i> passed to <code>main()</code> .

8.8.2.9 `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.

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

Outputs details for a user.

Parameters

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

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:

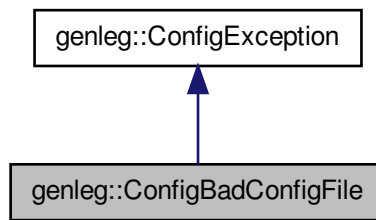
- lib/config/[config.h](#)
- lib/config/[config.cpp](#)
- lib/config/[config_getopt.cpp](#)

9.2 genleg::ConfigBadConfigFile Class Reference

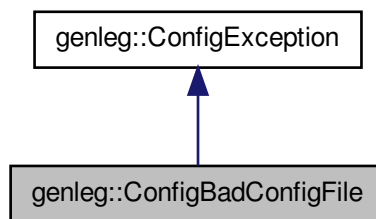
Exception class for badly formed configuration file.

```
#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:

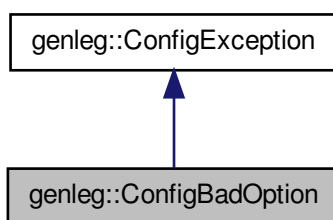
- [lib/config/config.h](#)

9.3 genleg::ConfigBadOption Class Reference

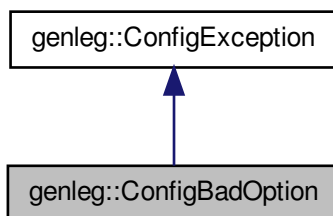
Exception class for bad provided option.

```
#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:

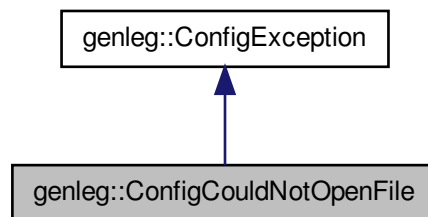
- `lib/config/config.h`

9.4 `genleg::ConfigCouldNotOpenFile` Class Reference

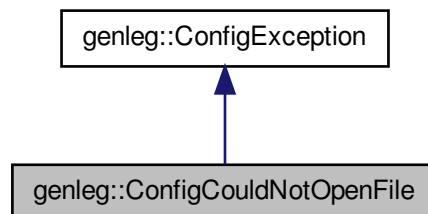
Exception class for when conf file cannot be opened.

```
#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:

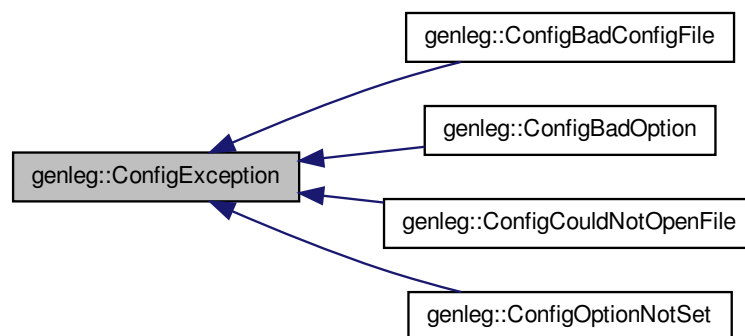
- [lib/config/config.h](#)

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:

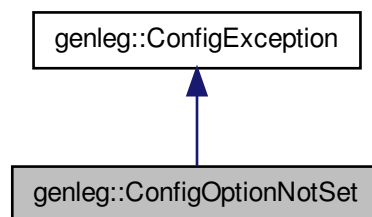
- `lib/config/config.h`

9.6 `genleg::ConfigOptionNotSet` Class Reference

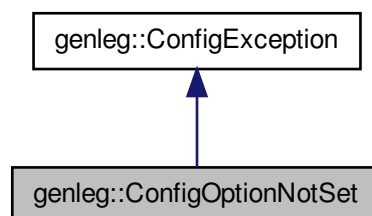
Exception class for option not set.

```
#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

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

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

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

9.7 pgutils::Currency Class Reference

[Currency](#) amount class.

```
#include <currency.h>
```

Public Member Functions

- [Currency](#) (const int64_t i=0, const uint8_t f=0)
Constructor.
- [Currency operator-](#) () const
Unary negation opertor.
- [Currency](#) & [operator+=](#) (const [Currency](#) &rhs)
Addition assignment operator.
- [Currency](#) & [operator-=](#) (const [Currency](#) &rhs)
Subtraction assignment operator.

Private Member Functions

- int64_t [expand](#) () const
Returns a [Currency](#) amount as a whole integer.

Private Attributes

- int64_t [m_int](#)
- int [m_frac](#)

Friends

- bool `operator==` (const [Currency](#) &lhs, const [Currency](#) &rhs)
[Currency](#) equality comparison operator.
- bool `operator<` (const [Currency](#) &lhs, const [Currency](#) &rhs)
[Currency](#) less than comparison operator.
- [Currency](#) `operator+` ([Currency](#) lhs, const [Currency](#) &rhs)
[Currency](#) addition operator.

9.7.1 Detailed Description

[Currency](#) amount class.

9.7.2 Constructor & Destructor Documentation

9.7.2.1 `pgutils::Currency::Currency (const int64_t i = 0, const uint8_t f = 0)` `[inline]`, `[explicit]`

Constructor.

Parameters

<i>i</i>	The integer part.
<i>f</i>	The fractional part.

9.7.3 Member Function Documentation

9.7.3.1 `int64_t pgutils::Currency::expand () const` `[inline]`, `[private]`

Returns a [Currency](#) amount as a whole integer.

Returns

The [Currency](#) amount expanded to a whole integer, equivalent to multiplying it by 100.

9.7.3.2 `Currency & Currency::operator+= (const Currency & rhs)`

Addition assignment operator.

Parameters

<i>rhs</i>	Right hand side currency amount.
------------	----------------------------------

Returns

A reference to the original currency amount.

9.7.3.3 `Currency pgutils::Currency::operator- () const` `[inline]`

Unary negation operator.

Returns

The negated currency amount.

9.7.3.4 Currency & Currency::operator-= (const Currency & rhs)

Subtraction assignment operator.

Parameters

<i>rhs</i>	Right hand side currency amount.
------------	----------------------------------

Returns

A reference to the original currency amount.

9.7.4 Friends And Related Function Documentation

9.7.4.1 Currency operator+ (Currency lhs, const Currency & rhs) [friend]

Currency addition operator.

Friend addition operator function

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Returns

The sum of the two sides.

9.7.4.2 bool operator< (const Currency & lhs, const Currency & rhs) [friend]

Currency less than comparison operator.

Friend less than comparison operator function

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Return values

<i>true</i>	If the lhs is less than the rhs.
<i>false</i>	If the lhs is not less than the rhs.

9.7.4.3 bool operator== (const Currency & lhs, const Currency & rhs) [friend]

Currency equality comparison operator.

Friend equality operator function

Parameters

<i>lhs</i>	Left hand side.
<i>rhs</i>	Right hand side.

Return values

<i>true</i>	If the two sides are equal.
<i>false</i>	If the two sides are not equal.

9.7.5 Member Data Documentation

9.7.5.1 `int pgutils::Currency::m_frac` `[private]`

Fractional part

9.7.5.2 `int64_t pgutils::Currency::m_int` `[private]`

Integer part

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

- [lib/pgutils/currency.h](#)
- [lib/pgutils/currency.cpp](#)

9.8 `pgutils::CurrencyException` Class Reference

Base [Currency](#) exception class.

```
#include <currency.h>
```

Public Member Functions

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

9.8.1 Detailed Description

Base [Currency](#) exception class.

9.8.2 Constructor & Destructor Documentation

9.8.2.1 `pgutils::CurrencyException::CurrencyException (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:

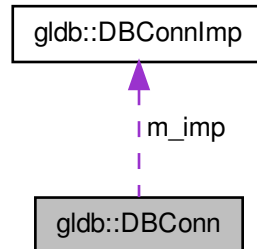
- [lib/pgutils/currency.h](#)

9.9 `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.
- unsigned long long [last_auto_increment](#) ()
Returns the last auto incremented value.
- [DBConn](#) (const [DBConn](#) &)
- [DBConn](#) (const [DBConn](#) &&)
- [DBConn](#) & [operator=](#) (const [DBConn](#) &)
- [DBConn](#) & [operator=](#) (const [DBConn](#) &&)

Private Attributes

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

9.9.1 Detailed Description

Database connection class.

9.9.2 Constructor & Destructor Documentation

9.9.2.1 DBConn::DBConn (DBConnImp * imp) [explicit]

Constructor.

Parameters

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

9.9.2.2 `gldb::DBConn::DBConn (const DBConn &)`

Deleted copy constructor

9.9.2.3 `gldb::DBConn::DBConn (const DBConn &&)`

Deleted move constructor

9.9.3 Member Function Documentation

9.9.3.1 `unsigned long long DBConn::last_auto_increment ()`

Returns the last auto incremented value.

Returns

The last auto incremented value.

9.9.3.2 `DBConn& gldb::DBConn::operator= (const DBConn &)`

Deleted copy assignment operator

9.9.3.3 `DBConn& gldb::DBConn::operator= (const DBConn &&)`

Deleted move assignment operator

9.9.3.4 `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.

9.9.3.5 `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.

9.9.4 Member Data Documentation

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

Pointer to database implementation object.

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

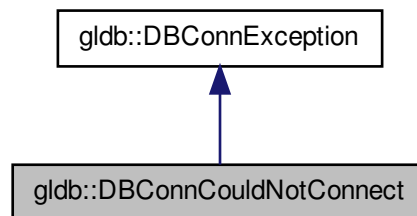
- lib/database/dbconn.h
- lib/database/dbconn.cpp

9.10 glldb::DBConnCouldNotConnect Class Reference

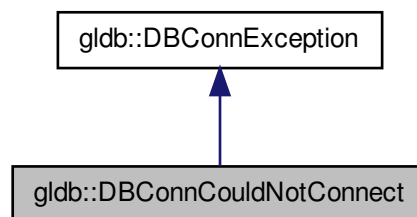
Could not connect to database exception class.

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

Inheritance diagram for glldb::DBConnCouldNotConnect:



Collaboration diagram for glldb::DBConnCouldNotConnect:



Public Member Functions

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

Constructor.

9.10.1 Detailed Description

Could not connect to database exception class.

9.10.2 Constructor & Destructor Documentation

9.10.2.1 `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:

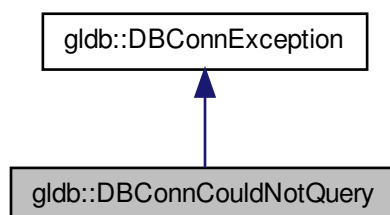
- [lib/database/dbconn.h](#)

9.11 gldb::DBConnCouldNotQuery Class Reference

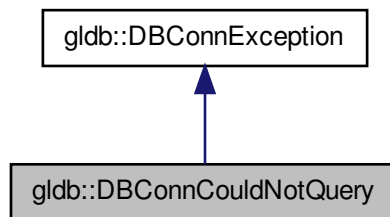
Could not execute database query exception class.

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

Inheritance diagram for `gldb::DBConnCouldNotQuery`:



Collaboration diagram for glldb::DBConnCouldNotQuery:



Public Member Functions

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

9.11.1 Detailed Description

Could not execute database query exception class.

9.11.2 Constructor & Destructor Documentation

9.11.2.1 `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:

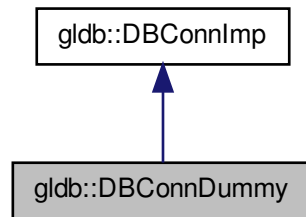
- lib/database/[dbconn.h](#)

9.12 glldb::DBConnDummy Class Reference

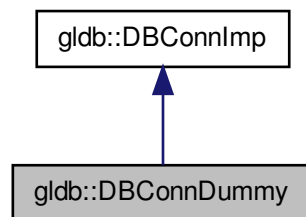
Dummy database implementation class.

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

Inheritance diagram for `gldb::DBConnDummy`:



Collaboration diagram for `gldb::DBConnDummy`:



Public Member Functions

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

Constructor.

- [DBConnDummy](#) (const [DBConnDummy](#) &)
- virtual [~DBConnDummy](#) ()
- [DBConnDummy](#) & [operator=](#) (const [DBConnDummy](#) &)
- 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.

9.12.1 Detailed Description

Dummy database implementation class.

9.12.2 Constructor & Destructor Documentation

9.12.2.1 `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.

9.12.2.2 `glldb::DBConnDummy::DBConnDummy (const DBConnDummy &)`

Deleted copy constructor

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

Destructor

9.12.3 Member Function Documentation

9.12.3.1 `DBConnDummy& glldb::DBConnDummy::operator= (const DBConnDummy &)`

Deleted assignment operator

9.12.3.2 `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](#).

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

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

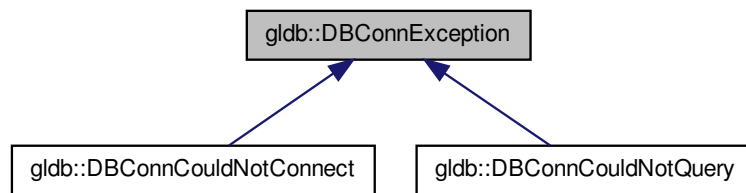
- [lib/database_imp/dummy/dbconn_dummy_imp.h](#)
- [lib/database_imp/dummy/dbconn_dummy_imp.cpp](#)

9.13 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.

9.13.1 Detailed Description

Base database connection exception class.

9.13.2 Constructor & Destructor Documentation

9.13.2.1 `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:

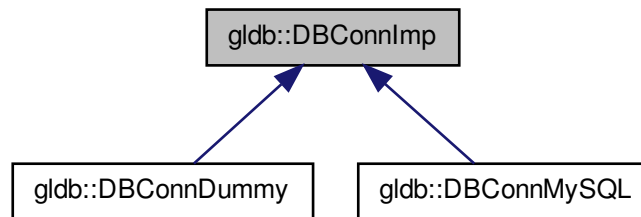
- [lib/database/dbconn.h](#)

9.14 gldb::DBConnImp Class Reference

Abstract database implementation base class.

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

Inheritance diagram for glldb::DBConnImp:



Public Member Functions

- [DBConnImp](#) ()
- virtual [~DBConnImp](#) ()
- virtual void [query](#) (const std::string &sql_query)=0
Runs an SQL query.
- virtual [Table select](#) (const std::string &query)=0
Runs an SQL SELECT query.
- virtual unsigned long long [last_auto_increment](#) ()=0
Returns the last auto incremented value.

9.14.1 Detailed Description

Abstract database implementation base class.

9.14.2 Constructor & Destructor Documentation

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

Constructor

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

Destructor

9.14.3 Member Function Documentation

9.14.3.1 `virtual unsigned long long glldb::DBConnImp::last_auto_increment () [pure virtual]`

Returns the last auto incremented value.

Returns

The last auto incremented value.

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

9.14.3.2 `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](#).

9.14.3.3 `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:

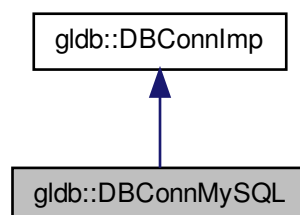
- [lib/database/dbconnimp.h](#)

9.15 gldb::DBConnMySQL Class Reference

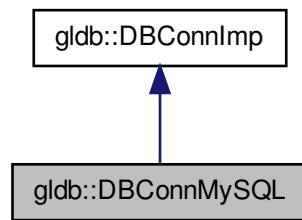
MySQL database implementation class.

```
#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.
- unsigned long long [last_auto_increment](#) ()
Returns the last auto incremented value.

Private Attributes

- MYSQL * [m_conn](#)
- std::mutex [m_mtx](#)

9.15.1 Detailed Description

MySQL database implementation class.

9.15.2 Constructor & Destructor Documentation

- 9.15.2.1 [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.
--	-----------------------------------

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

Deleted copy constructor

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

Delete move constructor

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

Destructor

9.15.3 Member Function Documentation

9.15.3.1 `unsigned long long gldb::DBConnMySQL::last_auto_increment () [virtual]`

Returns the last auto incremented value.

Returns

The last auto incremented value.

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

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

Deleted assignment operator

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

Deleted move assignment operator

9.15.3.4 `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](#).

9.15.3.5 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](#).

9.15.4 Member Data Documentation

9.15.4.1 MySQL* [gldb::DBConnMySQL::m_conn](#) [private]

The initialized MySQL handle.

9.15.4.2 std::mutex [gldb::DBConnMySQL::m_mtx](#) [private]

Database connection mutex

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

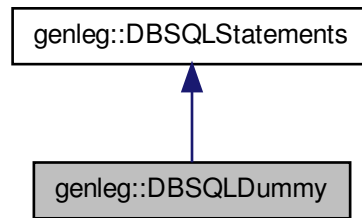
- lib/database_imp/mysql/[dbconn_mysql_imp.h](#)
- lib/database_imp/mysql/[dbconn_mysql_imp.cpp](#)

9.16 genleg::DBSQLDummy Class Reference

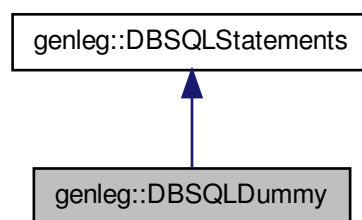
Dummy SQL statements class.

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

Inheritance diagram for genleg::DBSQLDummy:



Collaboration diagram for genleg::DBSQLDummy:



Additional Inherited Members

9.16.1 Detailed Description

Dummy SQL statements class.

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

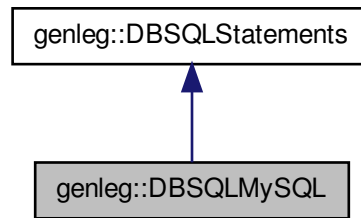
- lib/dbsql/[dbsql_dummy.h](#)

9.17 genleg::DBSQLMySQL Class Reference

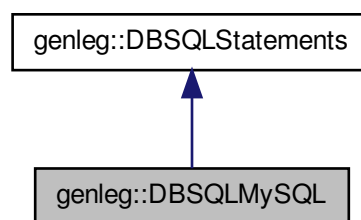
MySQL SQL statements class.

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


Inheritance diagram for genleg::DBSQLMySQL:



Collaboration diagram for genleg::DBSQLMySQL:



Additional Inherited Members

9.17.1 Detailed Description

MySQL SQL statements class.

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

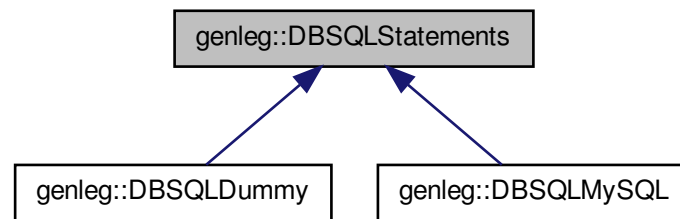
- lib/dbsql/[dbsql_mysql.h](#)

9.18 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 [post_je](#) (const unsigned int user, const unsigned int entity, const int period, const int year, const std::string &source, const std::string &memo) const
Returns a SQL INSERT statement to post a journal entry.
- virtual std::string [post_je_line](#) (const unsigned long long je, const std::string account, const std::string amount) const
Returns a SQL INSERT query to post a journal entry line.
- 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.
- virtual std::string [currenttb](#) () const
Returns a SQL statement to run the current trial balance report.
- virtual std::string [currenttb_by_entity](#) (const std::string &entity) const
Returns a SQL statement to run the current trial balance report by entity.
- std::string [listusers](#) () const
Returns a SQL statement to run the list users report.

9.18.1 Detailed Description

SQL statements class.

9.18.2 Constructor & Destructor Documentation

9.18.2.1 DBSQLStatements::DBSQLStatements ()

Constructor

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

Destructor

9.18.3 Member Function Documentation

9.18.3.1 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.

9.18.3.2 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.

9.18.3.3 std::string DBSQLStatements::currenttb () const [virtual]

Returns a SQL statement to run the current trial balance report.

Returns

The SQL statement.

9.18.3.4 std::string DBSQLStatements::currenttb_by_entity (const std::string & *entity*) const [virtual]

Returns a SQL statement to run the current trial balance report by entity.

Parameters

<i>entity</i>	The entity number for which to run the report.
---------------	--

Returns

The SQL statement.

9.18.3.5 `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.

9.18.3.6 `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.

9.18.3.7 `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.

9.18.3.8 `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.

9.18.3.9 `std::string DBSQLStatements::listusers () const`

Returns a SQL statement to run the list users report.

Returns

The SQL statement.

9.18.3.10 `std::string DBSQLStatements::post_je (const unsigned int user, const unsigned int entity, const int period, const int year, const std::string & source, const std::string & memo) const` [virtual]

Returns a SQL INSERT statement to post a journal entry.

Parameters

<i>user</i>	The ID of the posting user.
<i>entity</i>	The entity ID.
<i>period</i>	The accounting period.
<i>year</i>	The accounting year.
<i>source</i>	The journal entry source.
<i>memo</i>	The memo for the journal entry.

Returns

A string containing the query.

9.18.3.11 `std::string DBSQLStatements::post_je_line (const unsigned long long je, const std::string account, const std::string amount) const` [virtual]

Returns a SQL INSERT query to post a journal entry line.

Parameters

<i>je</i>	The journal entry ID.
<i>account</i>	The account to which to post.
<i>amount</i>	The amount to post.

Returns

A string containing the SQL statement.

9.18.3.12 `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.

9.18.3.13 `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.

9.18.3.14 `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.

9.18.3.15 `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:

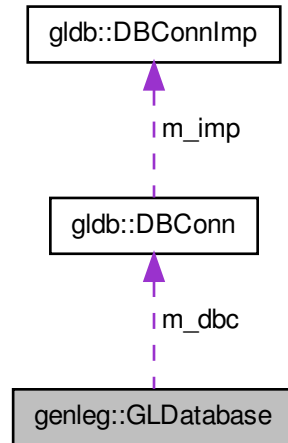
- lib/dbsql/[dbsqlstatements.h](#)
- lib/dbsql/[dbsqlstatements.cpp](#)

9.19 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.
- void [post_journal](#) (const [GLJournal](#) &journal)
Posts a journal entry.
- [GLReport](#) [report](#) (const std::string &report_name, const std::string &arg="")
Runs a report.

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.
- [GLReport current_trial_balance_report](#) (const std::string &entity)
Returns a current trial balance report.
- [GLReport list_users_report](#) ()
Returns a list users report.

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](#)

9.19.1 Detailed Description

General ledger database class.

9.19.2 Constructor & Destructor Documentation

- 9.19.2.1 [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.
-------------------------------	-----------

- 9.19.2.2 [GLDatabase::~~GLDatabase](#) ()

Destructor

9.19.3 Member Function Documentation

9.19.3.1 `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.

9.19.3.2 `void GLDatabase::create_structure ()`

Creates the database structure.

Exceptions

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

9.19.3.3 `GLUser GLDatabase::create_user (glldb::Table & table) [private]`

Creates a user from a query table.

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

Parameters

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

Returns

The new user.

9.19.3.4 `GLReport GLDatabase::current_trial_balance_report (const std::string & entity) [private]`

Returns a current trial balance report.

Parameters

<i>entity</i>	The entity for which to run the report, or an empty string for all entities.
---------------	--

Returns

A [*GLReport*](#) object with the report.

9.19.3.5 `void GLDatabase::destroy_structure ()`

Destroys the database structure.

Exceptions

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

9.19.3.6 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

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

9.19.3.7 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.
--------------------------------------	------------------------------

9.19.3.8 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.

9.19.3.9 GLReport GLDatabase::list_users_report () [private]

Returns a list users report.

Returns

A [*GLReport*](#) object with the report.

9.19.3.10 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.
--------------------------------------	-----------

9.19.3.11 void GLDatabase::post_journal (const GLJournal & *journal*)

Posts a journal entry.

Parameters

<i>journal</i>	The journal entry to post.
----------------	----------------------------

9.19.3.12 GLReport GLDatabase::report (const std::string & *report_name*, const std::string & *arg* = " ")

Runs a report.

Parameters

<i>report_name</i>	The name of the report.
<i>arg</i>	An optional argument.

Returns

A report object.

9.19.3.13 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.

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

Updates a user's details.

Parameters

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

9.19.4 Member Data Documentation

9.19.4.1 glldb::DBConn genleg::GLDatabase::m_dbc [private]

Database connection

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

SQL statements object

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

Vector containing database table names

9.19.4.4 `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](#)

9.20 genleg::GLDBException Class Reference

Base general ledger database exceptionc class.

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

Public Member Functions

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

9.20.1 Detailed Description

Base general ledger database exceptionc class.

9.20.2 Constructor & Destructor Documentation

9.20.2.1 `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:

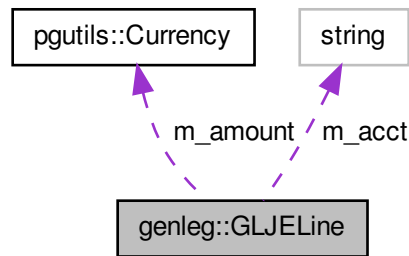
- [lib/gldb/glexception.h](#)

9.21 genleg::GLJELine Class Reference

Journal entry line class.

```
#include <gljournal.h>
```

Collaboration diagram for genleg::GLJELine:



Public Member Functions

- [GLJELine](#) (const std::string [account](#), const [pgutils::Currency](#) &[amount](#))
Constructor.
- std::string [account](#) () const
Returns the account name/number.
- [pgutils::Currency](#) [amount](#) () const
Returns the currency amount.

Private Attributes

- std::string [m_acct](#)
- [pgutils::Currency](#) [m_amount](#)

9.21.1 Detailed Description

Journal entry line class.

9.21.2 Member Function Documentation

9.21.2.1 std::string genleg::GLJELine::account () const [inline]

Returns the account name/number.

Returns

The account name/number.

9.21.2.2 pgutils::Currency genleg::GLJELine::amount () const [inline]

Returns the currency amount.

Returns

The currency amount.

9.21.3 Member Data Documentation

9.21.3.1 `std::string genleg::GLJELine::m_acct` `[private]`

Account number/name

9.21.3.2 `pgutils::Currency genleg::GLJELine::m_amount` `[private]`

Amount

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

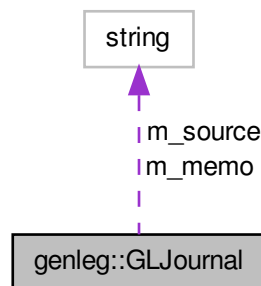
- [lib/gldb/gljournal.h](#)

9.22 `genleg::GLJournal` Class Reference

Journal entry class.

```
#include <gljournal.h>
```

Collaboration diagram for `genleg::GLJournal`:



Public Member Functions

- [GLJournal](#) (const unsigned int entity, const int period, const int year, const std::string &source, const std::string &memo)
Constructor.
- iterator [begin](#) ()
Returns an iterator to the first line.
- iterator [end](#) ()
Returns an iterator to one past the last line.
- const_iterator [begin](#) () const
Returns a const iterator to the first line.
- const_iterator [end](#) () const
Returns a const iterator to one past the last line.

Private Attributes

- unsigned int [m_entity](#)
- int [m_period](#)
- int [m_year](#)
- std::string [m_source](#)
- std::string [m_memo](#)
- std::vector< [GLJELine](#) > [m_lines](#)

9.22.1 Detailed Description

Journal entry class.

9.22.2 Constructor & Destructor Documentation

9.22.2.1 `genleg::GLJournal::GLJournal (const unsigned int entity, const int period, const int year, const std::string & source, const std::string & memo)` `[inline]`

Constructor.

Parameters

<i>entity</i>	The entity number.
<i>period</i>	The accounting period.
<i>year</i>	The accounting year.
<i>source</i>	The journal entry source.
<i>memo</i>	A memo for the journal entry.

9.22.3 Member Function Documentation

9.22.3.1 `iterator genleg::GLJournal::begin ()` `[inline]`

Returns an iterator to the first line.

Returns

An iterator to the first line.

9.22.3.2 `const_iterator genleg::GLJournal::begin () const` `[inline]`

Returns a const iterator to the first line.

Returns

A const iterator to the first line.

9.22.3.3 `iterator genleg::GLJournal::end ()` `[inline]`

Returns an iterator to one past the last line.

Returns

An iterator to one past the last line.

9.22.3.4 `const_iterator genleg::GLJournal::end () const` `[inline]`

Returns a const iterator to one past the last line.

Returns

A const iterator to one past the last line.

9.22.4 Member Data Documentation

9.22.4.1 `unsigned int genleg::GLJournal::m_entity` `[private]`

The entity number for the journal entry.

9.22.4.2 `std::vector<GLJELine> genleg::GLJournal::m_lines` `[private]`

A vector of journal entry lines.

9.22.4.3 `std::string genleg::GLJournal::m_memo` `[private]`

The memo for the journal entry.

9.22.4.4 `int genleg::GLJournal::m_period` `[private]`

The accounting period.

9.22.4.5 `std::string genleg::GLJournal::m_source` `[private]`

The journal entry source.

9.22.4.6 `int genleg::GLJournal::m_year` `[private]`

The accounting year.

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

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

9.23 `genleg::GLReport` Class Reference

General ledger report class.

```
#include <glreport.h>
```

Public Member Functions

- [GLReport](#) (const std::string &report)
- [~GLReport](#) ()

Private Attributes

- const std::string [m_report_text](#)

Friends

- `std::ostream & operator<< (std::ostream &out, const GLReport &report)`
Overridden << operator for printing a report.

9.23.1 Detailed Description

General ledger report class.

9.23.2 Constructor & Destructor Documentation

9.23.2.1 `genleg::GLReport::GLReport (const std::string & report)` `[inline]`

Constructor

9.23.2.2 `genleg::GLReport::~GLReport ()` `[inline]`

Destructor

9.23.3 Friends And Related Function Documentation

9.23.3.1 `std::ostream& operator<< (std::ostream & out, const GLReport & report)` `[friend]`

Overridden << operator for printing a report.

Parameters

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

Returns

A reference to `out`.

9.23.4 Member Data Documentation

9.23.4.1 `const std::string genleg::GLReport::m_report_text` `[private]`

The main report text

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

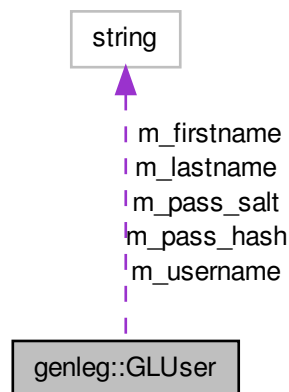
- `lib/gldb/glreport.h`

9.24 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](#)

9.24.1 Detailed Description

General ledger user class.

9.24.2 Constructor & Destructor Documentation

9.24.2.1 `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.

9.24.2.2 `GLUser::~~GLUser ()`

Destructor

9.24.3 Member Function Documentation

9.24.3.1 `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.

9.24.3.2 `bool GLUser::enabled () const`

Returns the user's enabled status.

Returns

The user's enabled status.

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

Returns the user's first name.

Returns

The user's first name.

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

Returns the user ID.

Returns

The user ID.

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

Returns the user's last name.

Returns

The user's last name.

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

Returns the user's hashed password.

Returns

The user's hashed password.

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

Returns the user's password salt.

Returns

The user's password salt.

9.24.3.8 `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.

9.24.3.9 `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.
--------------------	--------------------------------

9.24.3.10 `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.
----------------------	----------------------------

9.24.3.11 `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.
---------------------	---------------------------

9.24.3.12 `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.
-----------------	---

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

Sets a user's username.

Parameters

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

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

Returns the username.

Returns

The username.

9.24.4 Member Data Documentation

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

User's enabled status

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

User's first name

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

User ID

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

User's last name

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

User's hashed password

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

User's password salt

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

List of permissions

9.24.4.8 `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](#)

9.25 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](#)

9.25.1 Detailed Description

MySQL result structure class.

9.25.2 Constructor & Destructor Documentation

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

Constructor.

Parameters

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

Exceptions

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

9.25.2.2 gldb::MySQLResult::~~MySQLResult ()

Destructor

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

Deleted copy constructor

9.25.2.4 gldb::MySQLResult::MySQLResult (MySQLResult && result)

Deleted move constructor

9.25.3 Member Function Documentation

9.25.3.1 `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.

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

Deleted copy assignment operator

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

Deleted move assignment operator

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

Returns the MYSQL_RES pointer.

Returns

The MYSQL_RES pointer.

9.25.4 Member Data Documentation

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

The number of fields in the result set

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

The MYSQL_RES pointer

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

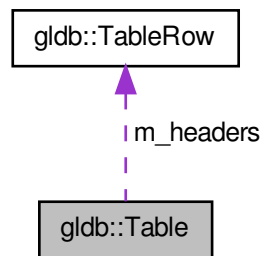
- [lib/database_imp/mysql/dbconn_mysql_result.h](#)
- [lib/database_imp/mysql/dbconn_mysql_result.cpp](#)

9.26 glldb::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.
- iterator [begin](#) ()
Returns iterator for beginning.
- iterator [end](#) ()
Returns iterator for end plus one.
- const_iterator [begin](#) () const
Returns const iterator for beginning.
- const_iterator [end](#) () const
Returns const iterator for end plus one.
- 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](#)

9.26.1 Detailed Description

Database table class.

9.26.2 Constructor & Destructor Documentation

9.26.2.1 [Table::Table](#) (const [TableRow](#) & *headers*) [explicit]

Constructor.

Parameters

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

9.26.2.2 [Table::Table](#) ([TableRow](#) && *headers*) [explicit]

Constructor with move semantics.

Parameters

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

9.26.2.3 [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.
--------------	--------------------------------

9.26.2.4 Table::Table (Table && *table*)

Move constructor.

Parameters

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

9.26.2.5 Table::~~Table ()

Destructor

9.26.3 Member Function Documentation

9.26.3.1 void Table::append_record (const TableRow & *new_record*)

Appends a record to the table.

Parameters

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

9.26.3.2 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.
-------------------	-----------------------

9.26.3.3 iterator glldb::Table::begin () [inline]

Returns iterator for beginning.

Returns

Iterator for beginning.

9.26.3.4 const_iterator glldb::Table::begin () const [inline]

Returns const iterator for beginning.

Returns

Const iterator for beginning.

9.26.3.5 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.

9.26.3.6 iterator glldb::Table::end () [inline]

Returns iterator for end plus one.

Returns

Iterator for end plus one.

9.26.3.7 const_iterator glldb::Table::end () const [inline]

Returns const iterator for end plus one.

Returns

Const iterator for end plus one.

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

9.26.3.9 `const TableRow& glldb::Table::get_headers () const [inline]`

Returns the field names.

Returns

The field names.

9.26.3.10 `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.

9.26.3.11 `size_t glldb::Table::num_fields () const [inline]`

Returns the number of fields in each row.

Returns

The number of fields in each row.

9.26.3.12 `size_t glldb::Table::num_records () const [inline]`

Returns the number of record in the table.

Returns

The number of records in the table.

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

Copy assignment operator.

Parameters

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

Returns

Reference to the assigned-to table.

9.26.3.14 `Table & Table::operator= (Table && table)`

Move assignment operator.

Parameters

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

Returns

Reference to the assigned-to table.

9.26.3.15 `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.

9.26.3.16 `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.
------------	---

9.26.3.17 `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.
------------	---

9.26.4 Member Data Documentation

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

The names of the fields

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

A vector to show if fields should be quoted for INSERT

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

A vector of the records

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

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

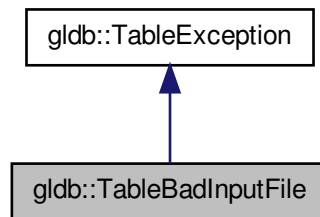
- [lib/database/table.cpp](#)

9.27 glldb::TableBadInputFile Class Reference

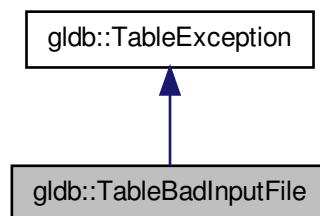
Could not connect to database exception class.

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

Inheritance diagram for glldb::TableBadInputFile:



Collaboration diagram for glldb::TableBadInputFile:



Public Member Functions

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

9.27.1 Detailed Description

Could not connect to database exception class.

9.27.2 Constructor & Destructor Documentation

9.27.2.1 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:

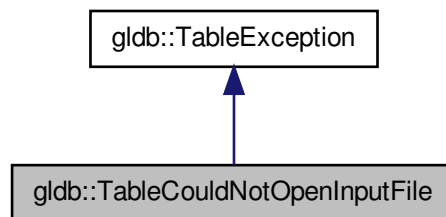
- lib/database/[table.h](#)

9.28 glldb::TableCouldNotOpenInputFile Class Reference

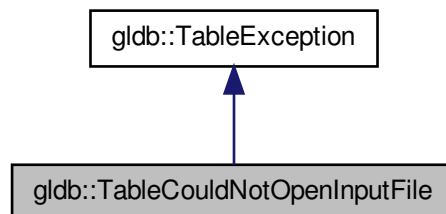
Could not connect to database exception class.

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

Inheritance diagram for glldb::TableCouldNotOpenInputFile:



Collaboration diagram for glldb::TableCouldNotOpenInputFile:



Public Member Functions

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

9.28.1 Detailed Description

Could not connect to database exception class.

9.28.2 Constructor & Destructor Documentation

9.28.2.1 `glldb::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:

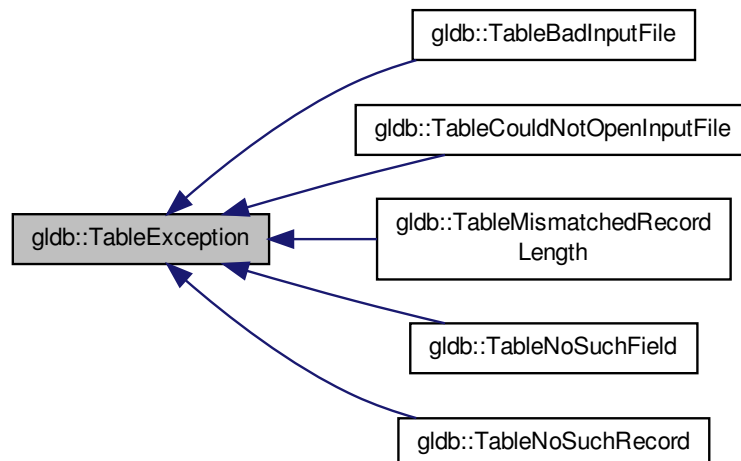
- [lib/database/table.h](#)

9.29 glldb::TableException Class Reference

Base database connection exception class.

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

Inheritance diagram for `glldb::TableException`:



Public Member Functions

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

Constructor.

9.29.1 Detailed Description

Base database connection exception class.

9.29.2 Constructor & Destructor Documentation

9.29.2.1 `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:

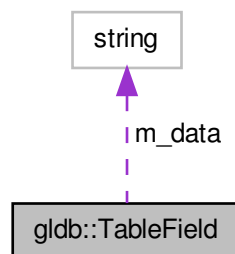
- [lib/database/table.h](#)

9.30 `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.

9.30.1 Detailed Description

Database table field class.

9.30.2 Constructor & Destructor Documentation

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

Constructor accepting `const char * data`.

Parameters

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

9.30.2.2 TableField::TableField (const std::string & data)

Constructor accepting `std::string data`.

Parameters

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

9.30.2.3 `TableField::TableField (std::string && data)`

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

Parameters

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

9.30.2.4 `TableField::TableField (const TableField & field)`

Copy constructor.

Parameters

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

9.30.2.5 `TableField::TableField (TableField && field)`

Move constructor.

Parameters

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

9.30.2.6 `TableField::~~TableField ()`

Destructor

9.30.3 Member Function Documentation

9.30.3.1 `size_t glldb::TableField::length () const` `[inline]`

Returns the length of the field.

Returns

The length of the field.

9.30.3.2 `glldb::TableField::operator std::string () const` `[inline]`

Overridden conversion operator.

Returns the field contents as a string.

9.30.3.3 `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.

9.30.3.4 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.

9.30.3.5 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.

9.30.3.6 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.

9.30.3.7 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.

9.30.3.8 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.

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

Overridden move assignment operator.

Parameters

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

Returns

A reference to the same field.

9.30.3.10 char& glldb::TableField::operator[] (const size_t *idx*) [inline]

Overridden index operator.

Parameters

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

Returns

A reference to the character at the specified index.

9.30.3.11 const char& glldb::TableField::operator[] (const size_t *idx*) const [inline]

Overridden index operator.

Parameters

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

Returns

A const reference to the character at the specified index.

9.30.4 Friends And Related Function Documentation

9.30.4.1 `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`.

9.30.5 Member Data Documentation

9.30.5.1 `std::string glldb::TableField::m_data` [*private*]

The field contents

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

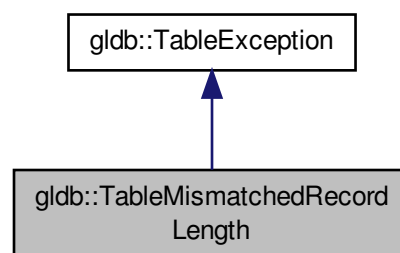
- [lib/database/tablefield.h](#)
- [lib/database/tablefield.cpp](#)

9.31 glldb::TableMismatchedRecordLength Class Reference

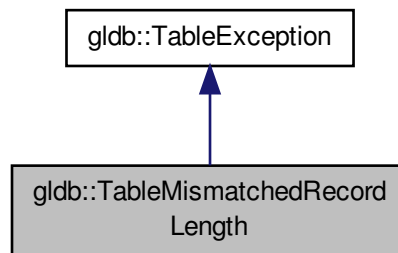
Mismatched record length exception class.

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

Inheritance diagram for `glldb::TableMismatchedRecordLength`:



Collaboration diagram for `gldb::TableMismatchedRecordLength`:



Public Member Functions

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

9.31.1 Detailed Description

Mismatched record length exception class.

9.31.2 Constructor & Destructor Documentation

9.31.2.1 `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:

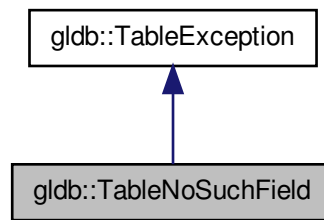
- lib/database/[table.h](#)

9.32 gldb::TableNoSuchField Class Reference

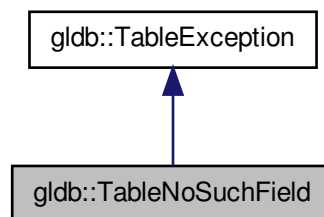
No such field exception class.

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


Inheritance diagram for gldb::TableNoSuchField:



Collaboration diagram for gldb::TableNoSuchField:



Public Member Functions

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

9.32.1 Detailed Description

No such field exception class.

9.32.2 Constructor & Destructor Documentation

9.32.2.1 `gldb::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:

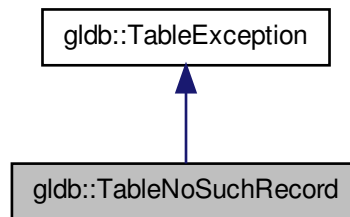
- [lib/database/table.h](#)

9.33 glDb::TableNoSuchRecord Class Reference

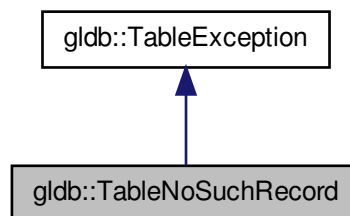
No such record exception class.

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

Inheritance diagram for glDb::TableNoSuchRecord:



Collaboration diagram for glDb::TableNoSuchRecord:



Public Member Functions

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

9.33.1 Detailed Description

No such record exception class.

9.33.2 Constructor & Destructor Documentation

9.33.2.1 glldb::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](#)

9.34 glldb::TableRow Class Reference

Database table row class.

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

Public Member Functions

- [TableRow](#) ()
- [TableRow](#) (const size_t size)
Constructor with initial number of fields.
- [TableRow](#) (const std::vector< std::string > &vec)
Constructor with string vector.
- [TableRow](#) (std::vector< std::string > &&vec)
Constructor with string vector and move semantics.
- [TableRow](#) (std::initializer_list< std::string > i)
Constructor with std::string initializer list.
- [TableRow](#) (const [TableRow](#) &row)
Copy constructor.
- [TableRow](#) ([TableRow](#) &&row)
Move constructor.
- [TableRow](#) & operator= (const [TableRow](#) &row)
Copy assignment operator.
- [TableRow](#) & operator= ([TableRow](#) &&row)
Move assignment operator.
- [~TableRow](#) ()
- size_t size () const
Returns the number of fields.
- iterator [begin](#) ()
Returns iterator for beginning.
- iterator [end](#) ()
Returns iterator for end plus one.
- const_iterator [begin](#) () const
Returns const iterator for beginning.
- const_iterator [end](#) () const
Returns const iterator for end plus one.
- [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`

9.34.1 Detailed Description

Database table row class.

9.34.2 Constructor & Destructor Documentation

9.34.2.1 `TableRow::TableRow ()`

Default constructor

9.34.2.2 `TableRow::TableRow (const size_t size)` `[explicit]`

Constructor with initial number of fields.

Parameters

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

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

Constructor with string vector.

Parameters

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

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

Constructor with string vector and move semantics.

Parameters

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

9.34.2.5 TableRow::TableRow (std::initializer_list< std::string > *i*) [explicit]

Constructor with std::string initializer list.

Parameters

<i>i</i>	The initializer list.
----------	-----------------------

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

Copy constructor.

Parameters

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

9.34.2.7 TableRow::TableRow (TableRow && *row*)

Move constructor.

Parameters

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

9.34.2.8 TableRow::~~TableRow ()

Destructor

9.34.3 Member Function Documentation

9.34.3.1 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.
------------------	--------------------------------

9.34.3.2 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.
------------------	--------------------------------

9.34.3.3 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.
------------------	--------------------------------

9.34.3.4 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.
------------------	-----------------------------

9.34.3.5 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.
------------------	-----------------------------

9.34.3.6 iterator glldb::TableRow::begin () [inline]

Returns iterator for beginning.

Returns

Iterator for beginning.

9.34.3.7 const_iterator glldb::TableRow::begin () const [inline]

Returns const iterator for beginning.

Returns

Const iterator for beginning.

9.34.3.8 iterator glldb::TableRow::end () [inline]

Returns iterator for end plus one.

Returns

Iterator for end plus one.

9.34.3.9 const_iterator glldb::TableRow::end () const [inline]

Returns const iterator for end plus one.

Returns

Const iterator for end plus one.

9.34.3.10 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.

9.34.3.11 TableRow & TableRow::operator= (TableRow && row)

Move assignment operator.

Parameters

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

Returns

A reference to the assigned-to row.

9.34.3.12 TableField& gldb::TableRow::operator[] (const size_t idx) [inline]

Overridden index operator.

Parameters

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

Returns

A reference to the field at the specified index.

9.34.3.13 const TableField& gldb::TableRow::operator[] (const size_t idx) const [inline]

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.

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

Prints a row.

Parameters

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

9.34.3.15 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.

9.34.3.16 std::string TableRow::record_string () const

Creates an unquoted comma separated string of fields.

Returns

The unquoted comma separated string.

9.34.3.17 size_t glldb::TableRow::size () const [inline]

Returns the number of fields.

Returns

The number of fields.

9.34.4 Member Data Documentation

9.34.4.1 std::vector<TableField> glldb::TableRow::m_fields [private]

A vector of fields

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

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

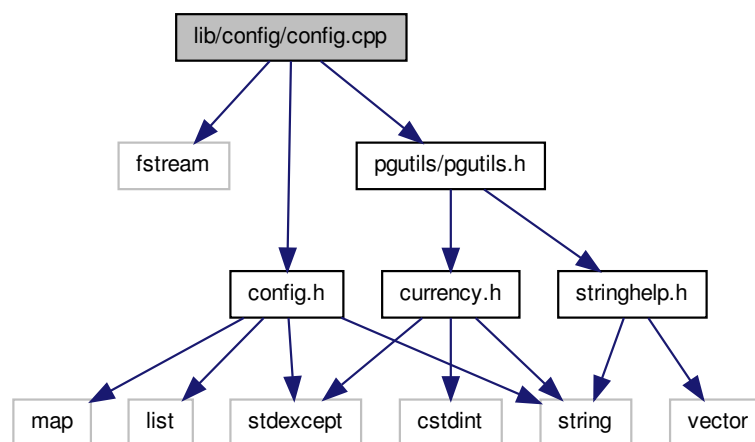
Chapter 10

File Documentation

10.1 lib/config/config.cpp File Reference

Implementation of program configurations class.

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



10.1.1 Detailed Description

Implementation of program configurations class.

Author

Paul Griffiths

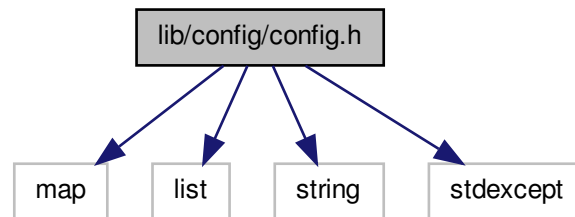
Copyright

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

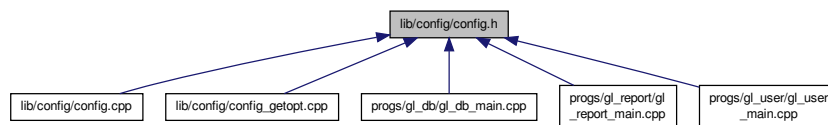
10.2 lib/config/config.h File Reference

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

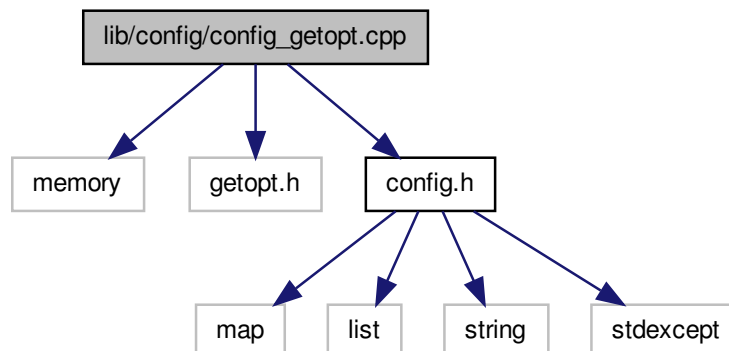
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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`

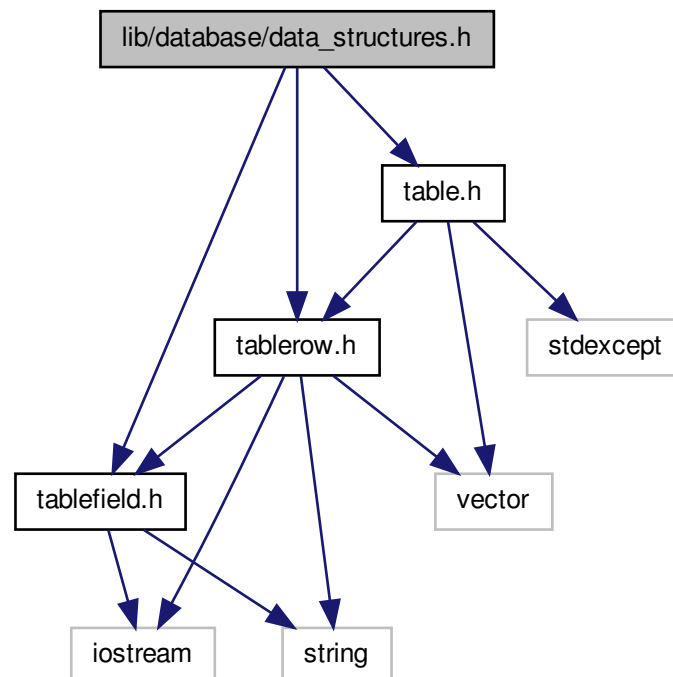
UNIX feature test macro for getopt library

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`:



Main interface to database data structures.

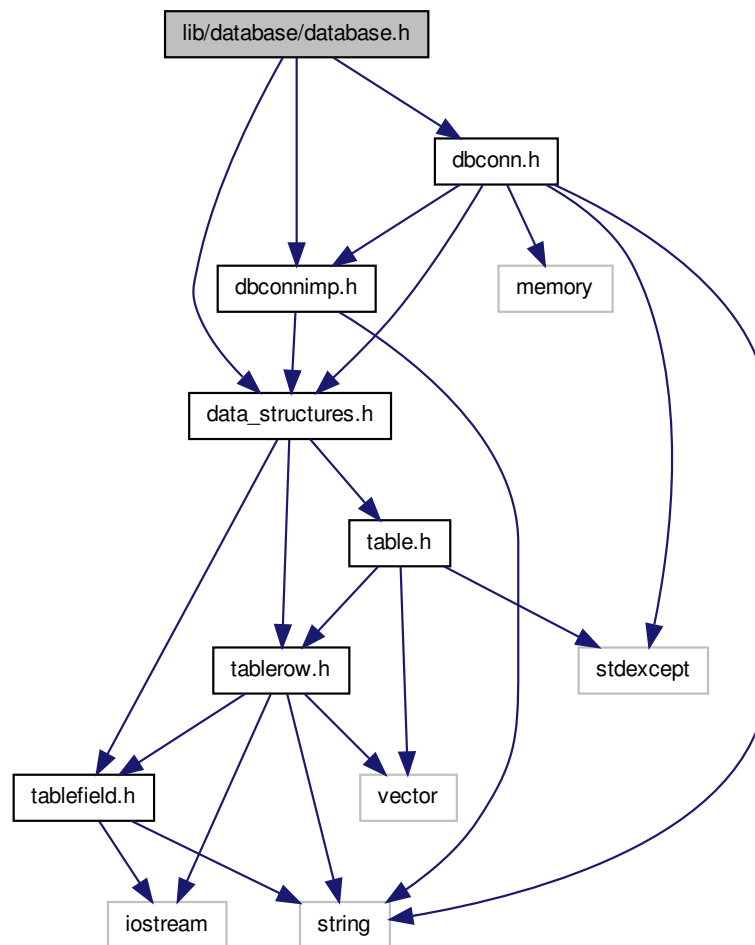
Paul Griffiths

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

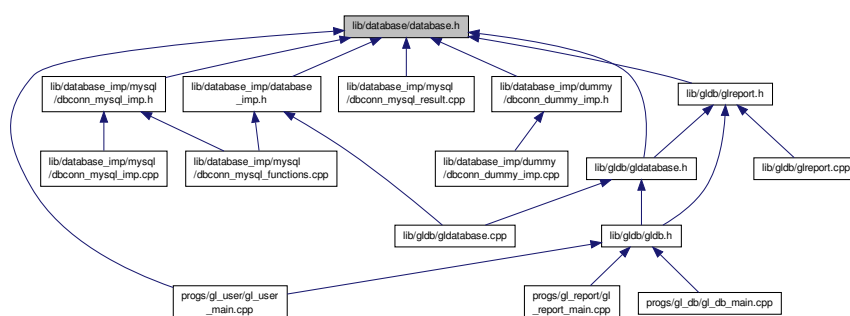
User interface to database functionality.

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

Include dependency graph for database.h:



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



10.5.1 Detailed Description

User interface to database functionality.

Author

Paul Griffiths

Copyright

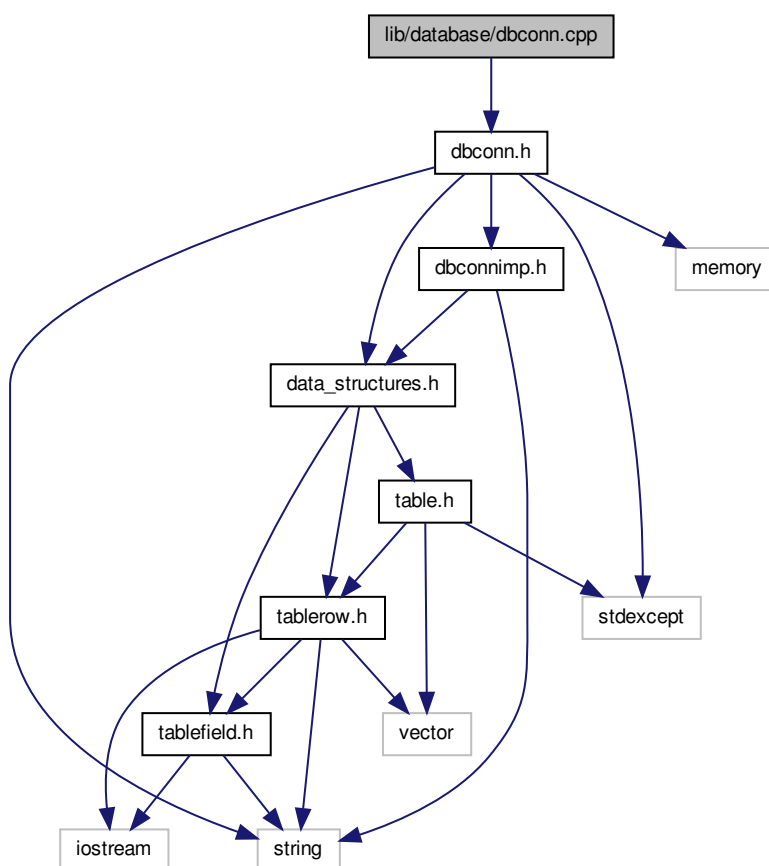
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

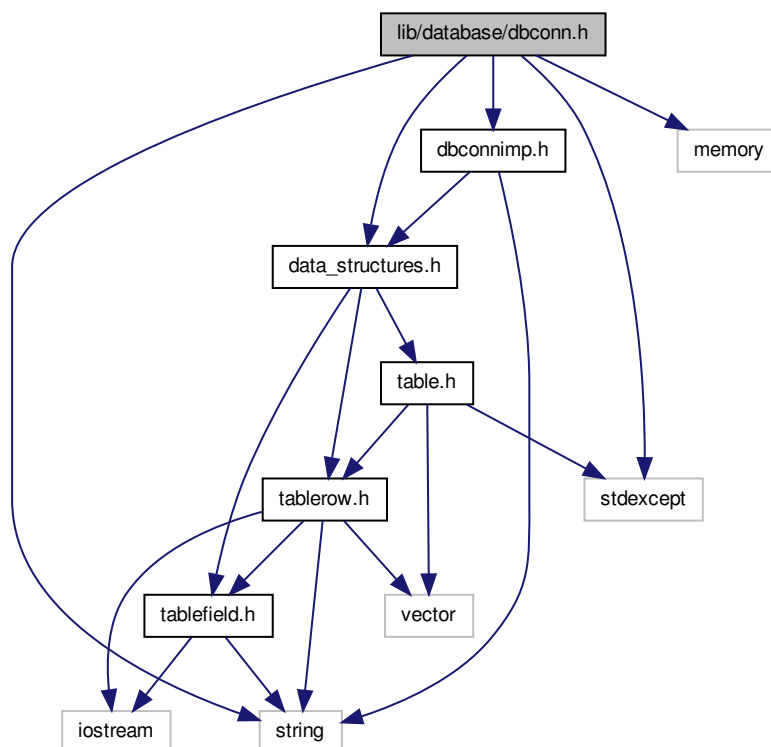
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.7 lib/database/dbconn.h File Reference

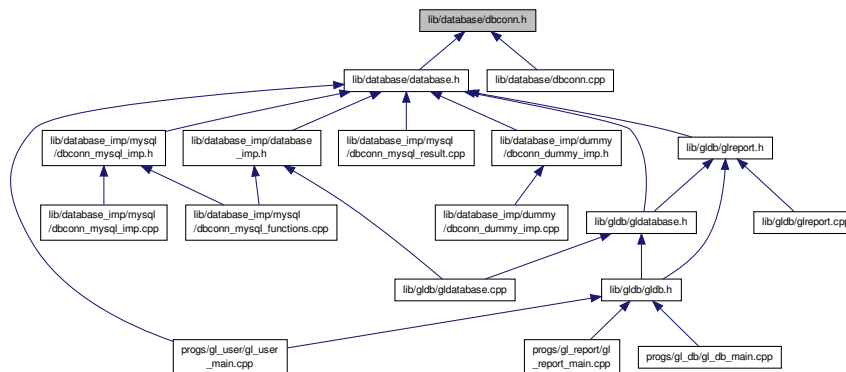
Interface to database connection base class.

```
#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"
```


Classes

- class [gldb::DBConnImp](#)
Abstract database implementation base class.

10.8.1 Detailed Description

Interface to abstract database implementation base class.

Author

Paul Griffiths

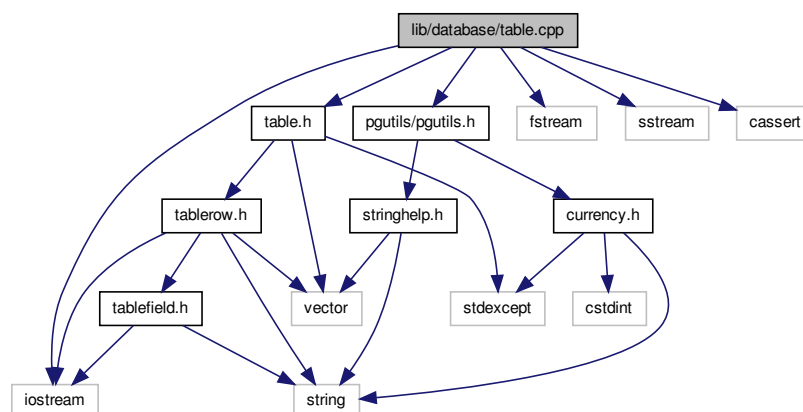
Copyright

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 "pgutils/pgutils.h"
Include dependency graph for table.cpp:
```



10.9.1 Detailed Description

Implementation of database table data structure.

Author

Paul Griffiths

Copyright

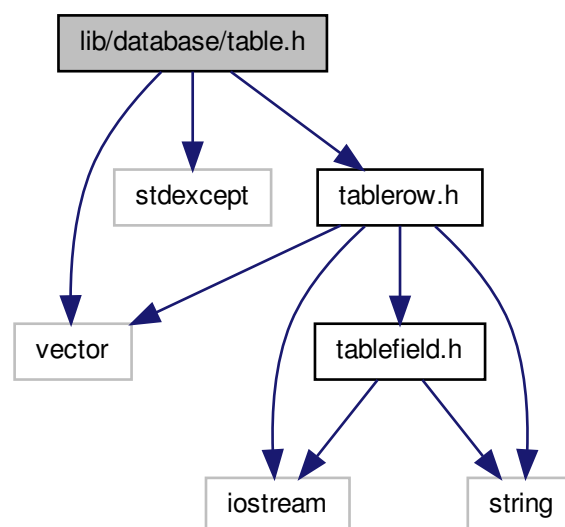
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.10 lib/database/table.h File Reference

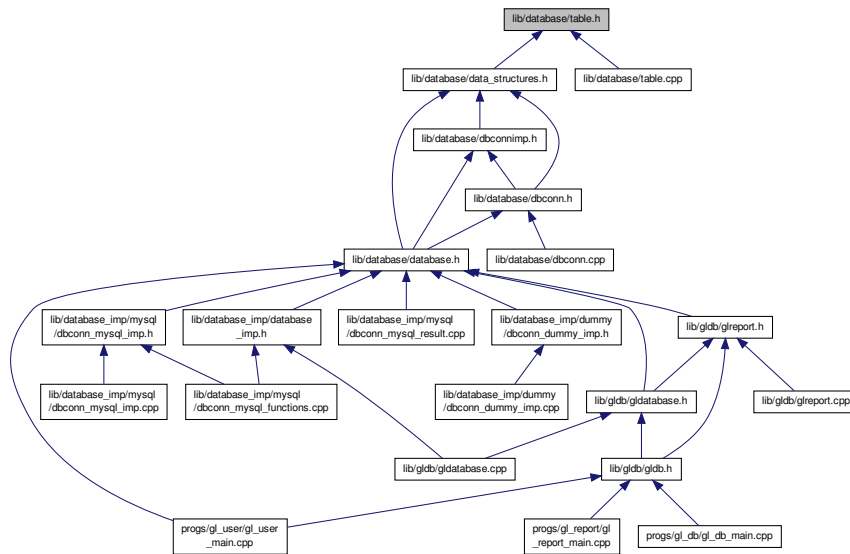
Interface to database table data structure.

```
#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

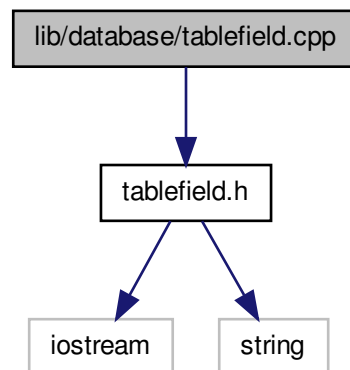
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.11 lib/database/tablefield.cpp File Reference

Implementation of database table field class.

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

Include dependency graph for tablefield.cpp:



10.11.1 Detailed Description

Implementation of database table field class.

Author

Paul Griffiths

Copyright

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

10.12 lib/database/tablefield.h File Reference

Interface to database table field class.

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

```
graph TD; A[lib/database/tablefield.h] --> B[iostream]; A --> C[string];
```

A diagram illustrating a dependency. A box labeled `lib/database/tablefield.h` has two arrows pointing down to boxes labeled `iostream` and `string`.

[illegible]

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

- `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

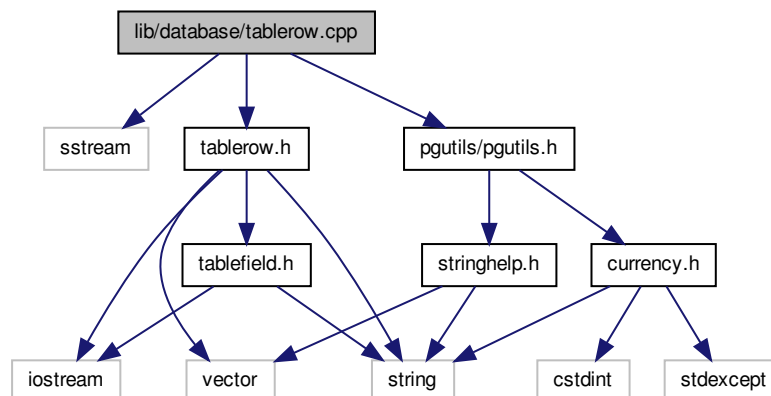
Copyright

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

10.13 lib/database/tablerow.cpp File Reference

Implementation of database table row data structure.

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



10.13.1 Detailed Description

Implementation of database table row data structure.

Author

Paul Griffiths

Copyright

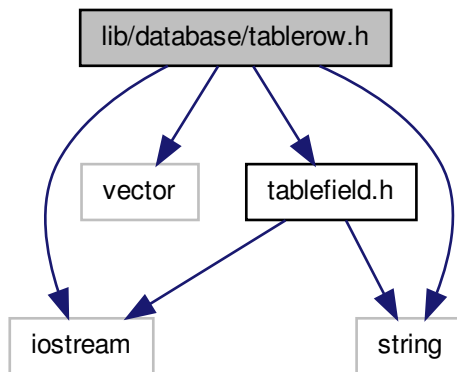
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.14 lib/database/tablerow.h File Reference

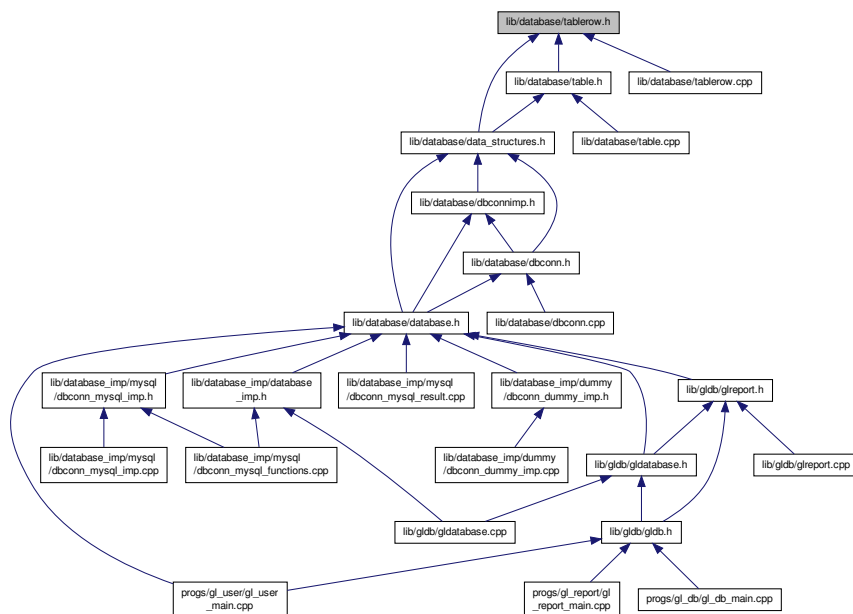
Interface to database table row data structure.

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

Include dependency graph for tablerow.h:



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



Classes

- class `gldb::TableRow`

Database table row class.

10.14.1 Detailed Description

Interface to database table row data structure.

Author

Paul Griffiths

Copyright

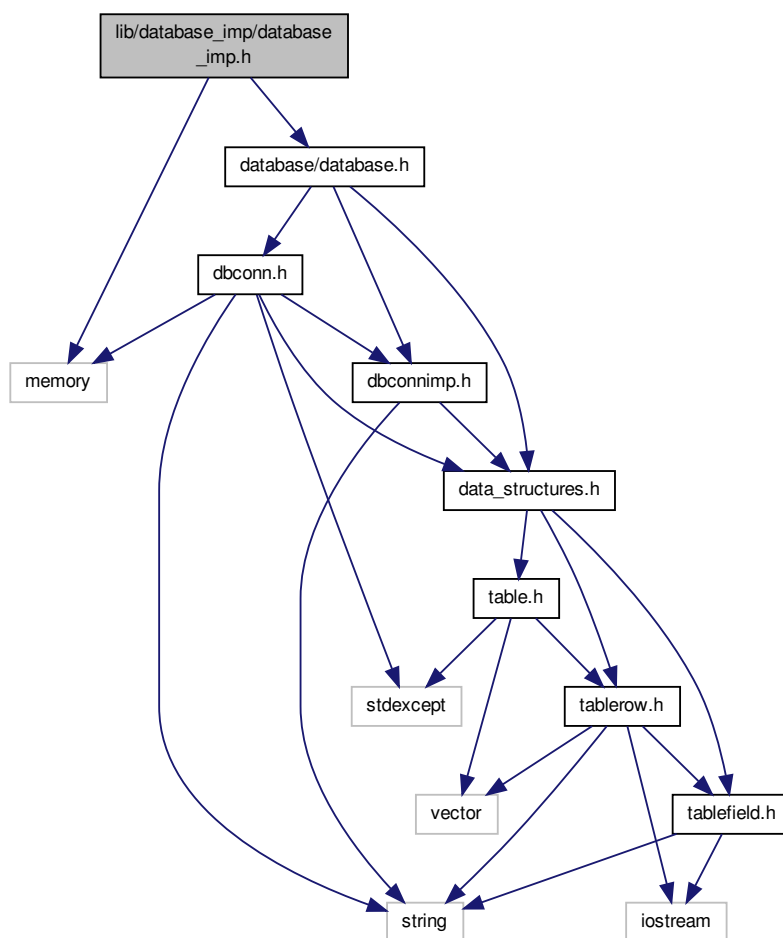
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.15 lib/database_imp/database_imp.h File Reference

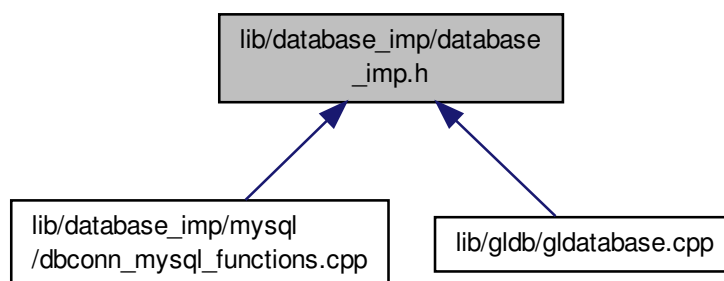
Interface to database implementation factory function.

```
#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 * glldb::get_connection` (const std::string &database, const std::string &hostname, const std::string &username, const std::string &password)

Creates and returns a pointer to a database implementation.

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

Returns the name of the compiled-in database type.

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"
```

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

Implementation of Dummy database connection implementation class.

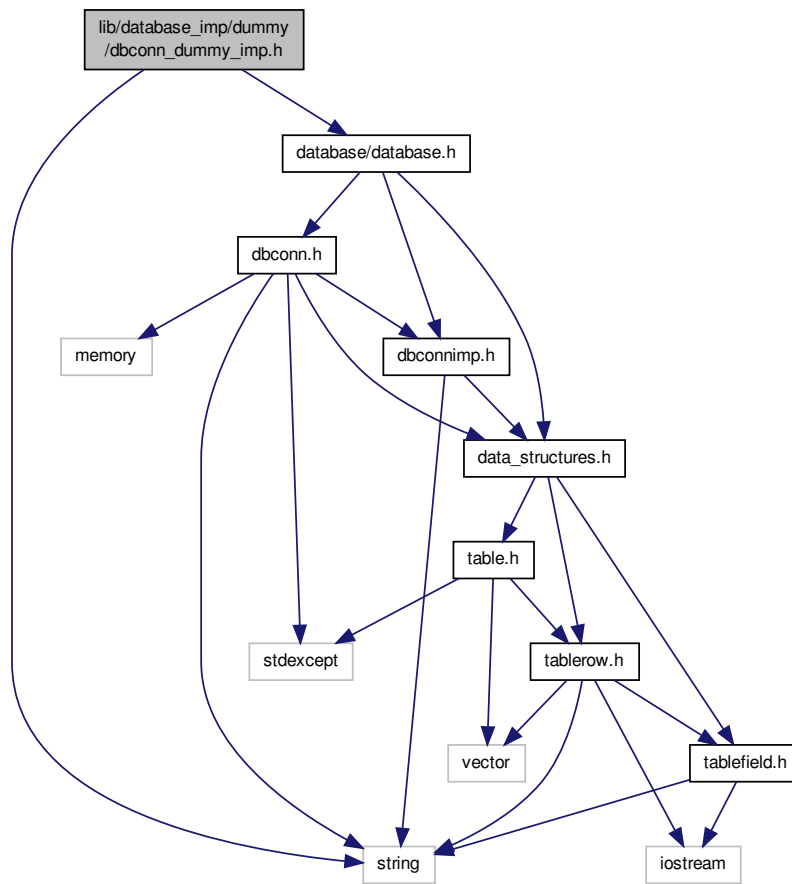
Paul Griffiths

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

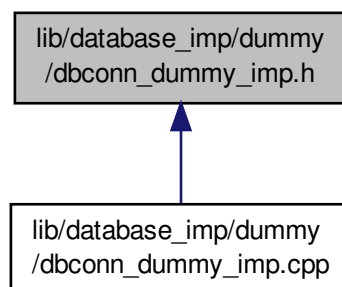
Interface to dummy database connection implementation class.

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

Include dependency graph for dbconn_dummy_imp.h:



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



Classes

- class `gldb::DBConnDummy`

Dummy database implementation class.

10.17.1 Detailed Description

Interface to dummy database connection implementation class.

Author

Paul Griffiths

Copyright

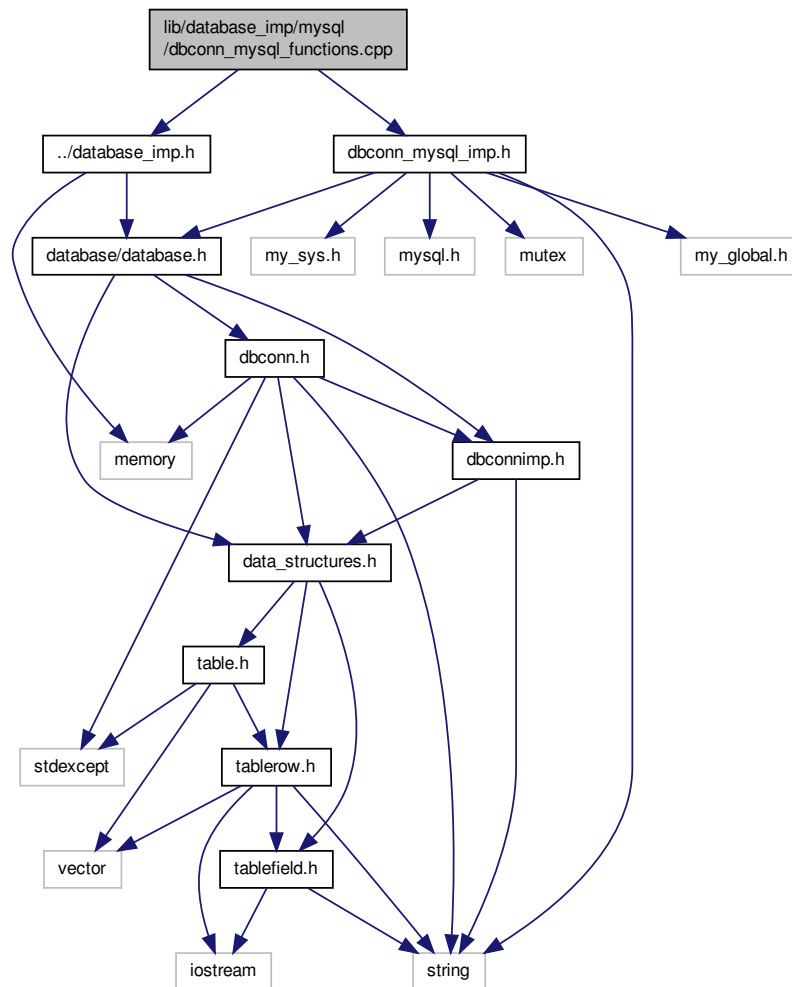
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.18 lib/database_imp/mysql/dbconn_mysql_functions.cpp File Reference

Implementation of MySQL implementation factory function.

```
#include "../database_imp.h"
#include "dbconn_mysql_imp.h"
```

Include dependency graph for `dbconn_mysql_functions.cpp`:



10.18.1 Detailed Description

Implementation of MySQL implementation factory function.

Author

Paul Griffiths

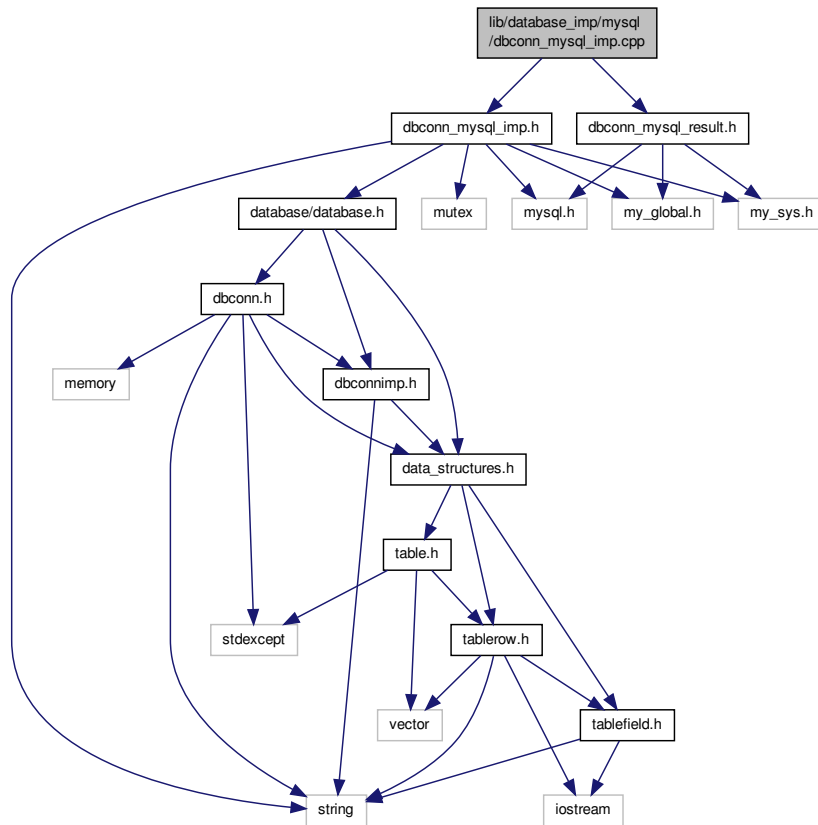
Copyright

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.

Author

Paul Griffiths

Copyright

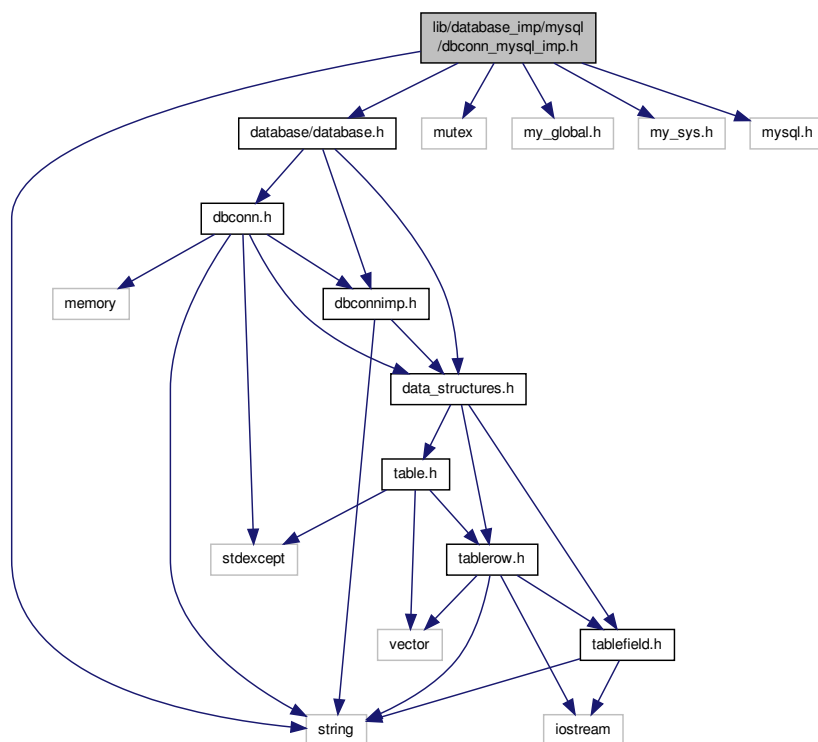
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.20 lib/database_imp/mysql/dbconn_mysql_imp.h File Reference

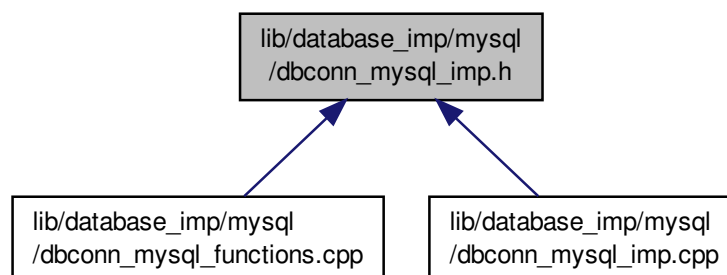
Interface to MySQL database connection implementation class.

```
#include <string>
#include <mutex>
#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

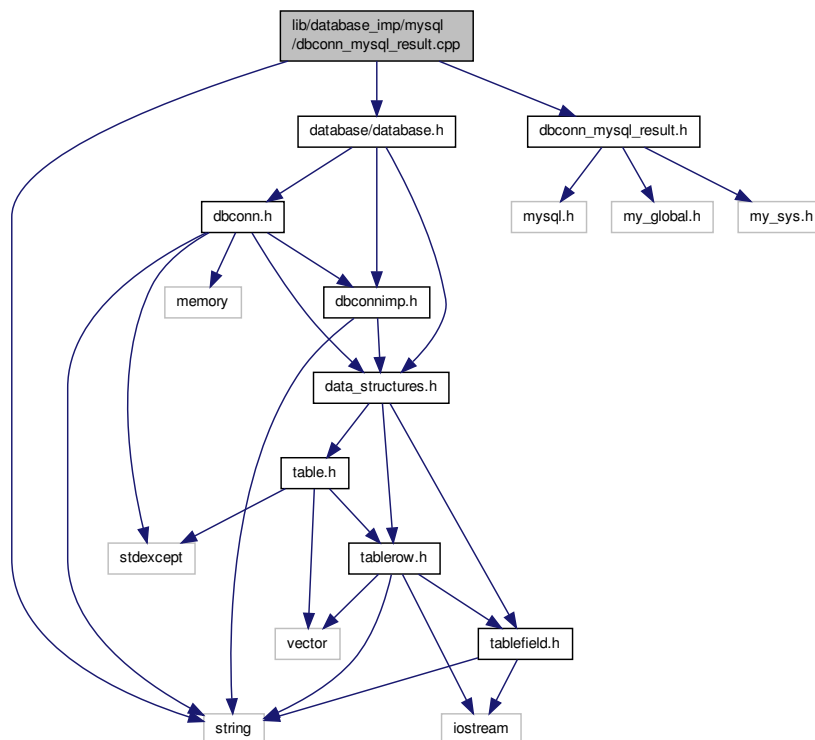
Copyright

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

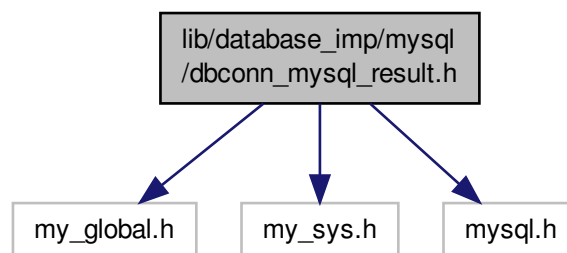
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

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

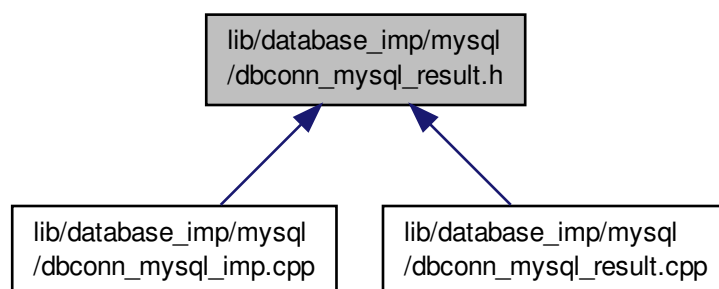
Interface to MySQL result structure resource handle class.

```
#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 `gldb::MySQLResult`
MySQL result structure class.

10.22.1 Detailed Description

Interface to MySQL result structure resource handle class.

Author

Paul Griffiths

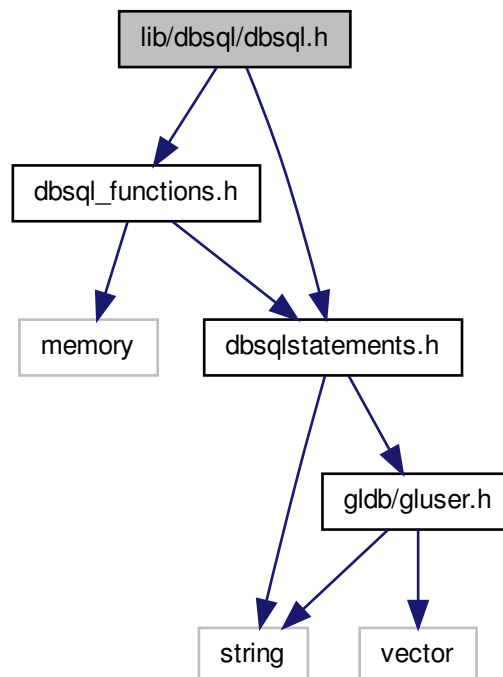
Copyright

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

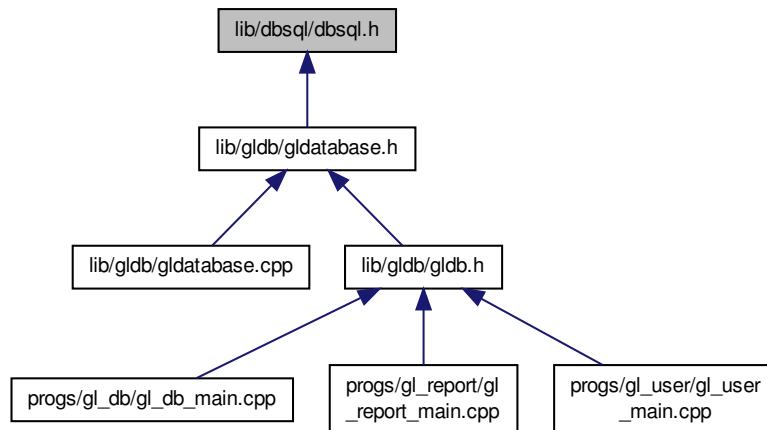
10.23 lib/dbsql/dbsql.h File Reference

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

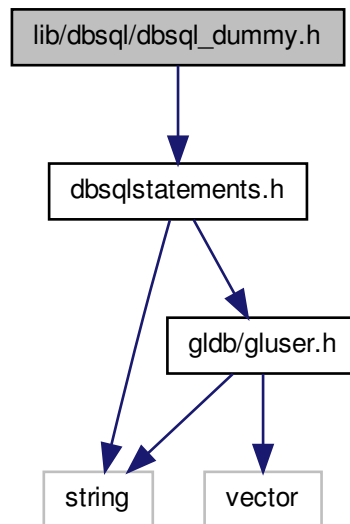
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.24 lib/dbsql/dbsql_dummy.h File Reference

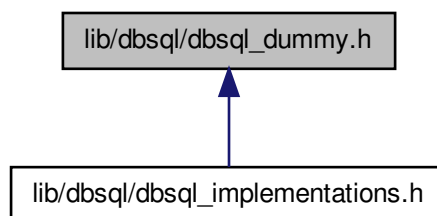
Interface to dummy SQL statement class.

```
#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/>

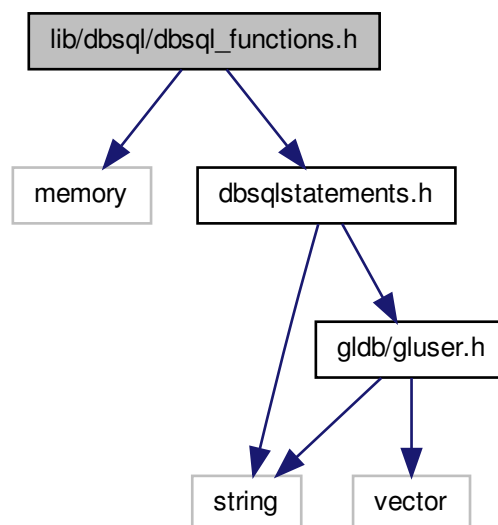
10.25 lib/dbsql/dbsql_functions.h File Reference

Interface to SQL module standalone functions.

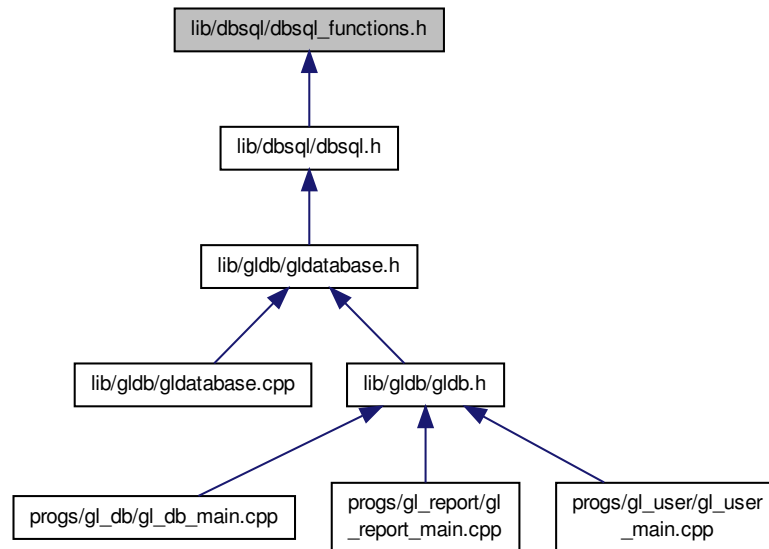
```
#include <memory>
```

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

Include dependency graph for `dbsql_functions.h`:



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



Functions

- `std::shared_ptr< const DBSQLStatements > genleg::get_sql_object ()`
Factory function for DBSQL objects.

10.25.1 Detailed Description

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/>

10.26 lib/dbsql/dbsql_implementations.h File Reference

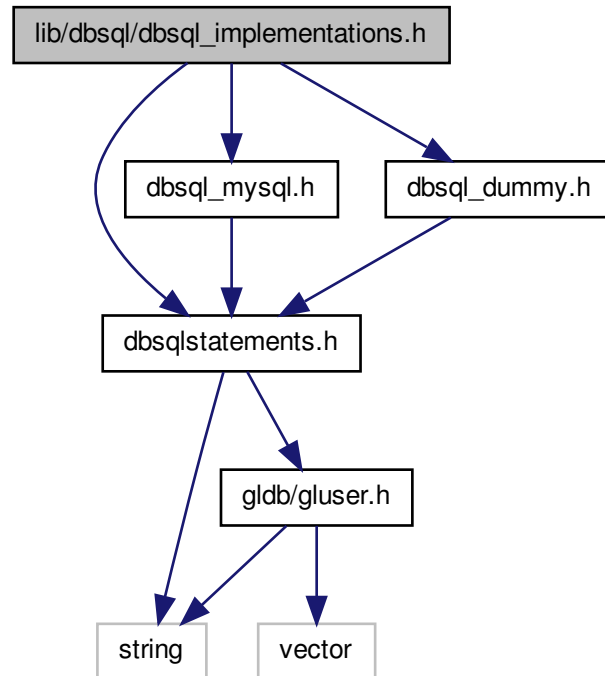
Aggregation header for DBSQLStatements implementations.

```

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

```

Include dependency graph for `dbssl_implementations.h`:



10.26.1 Detailed Description

Aggregation header for DBSqlStatements implementations.

Author

Paul Griffiths

Copyright

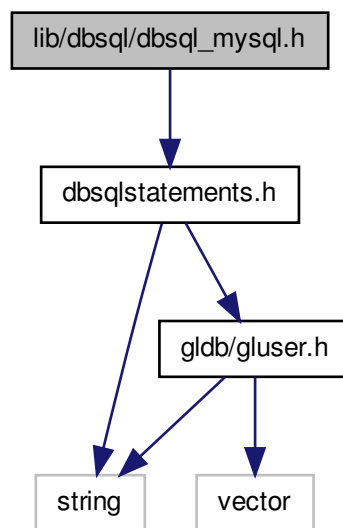
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.27 lib/dbsql/dbsql_mysql.h File Reference

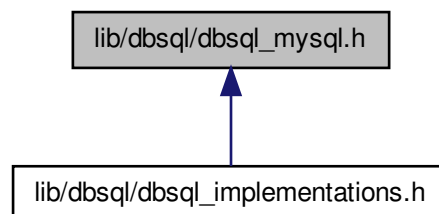
Interface to MySQL SQL statement class.

```
#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.

10.27.1 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/>

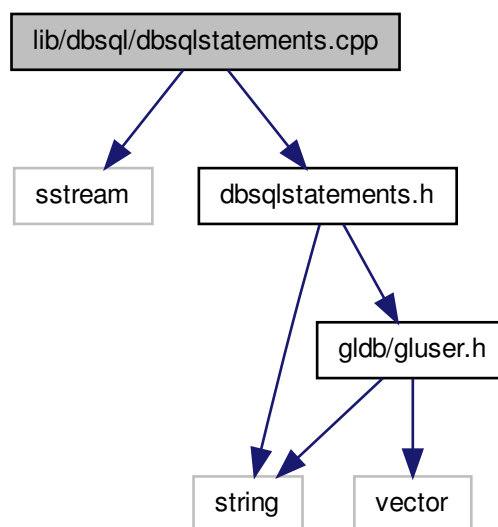
10.28 lib/dbsql/dbsqlstatements.cpp File Reference

Implementation of SQL statement class.

```
#include <sstream>
```

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

Include dependency graph for dbsqlstatements.cpp:



10.28.1 Detailed Description

Implementation of SQL statement class.

Author

Paul Griffiths

Copyright

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

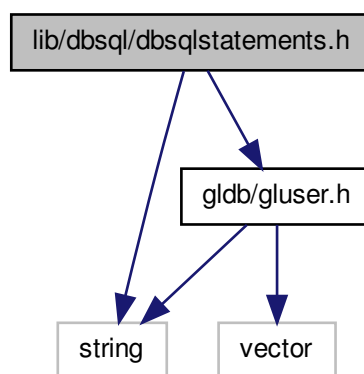
10.29 lib/dbsql/dbsqlstatements.h File Reference

Implementation of SQL module standalone functions.

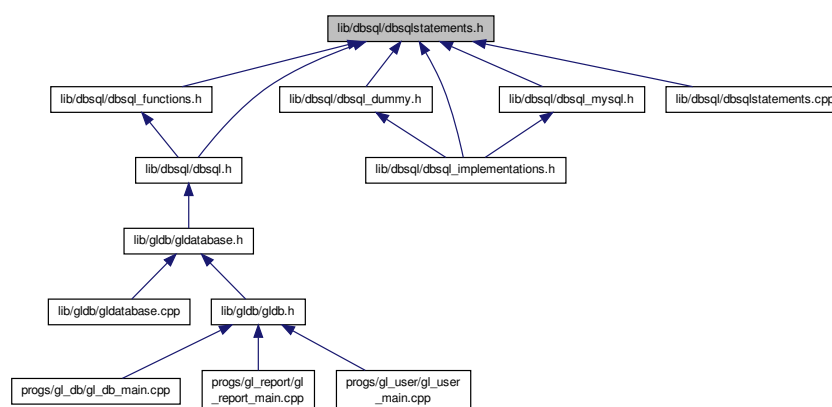
```
#include <string>
```

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

Include dependency graph for dbsqlstatements.h:



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



Classes

- class `genleg::DBSQLStatements`
SQL statements class.

10.29.1 Detailed Description

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

Author

Paul Griffiths

Copyright

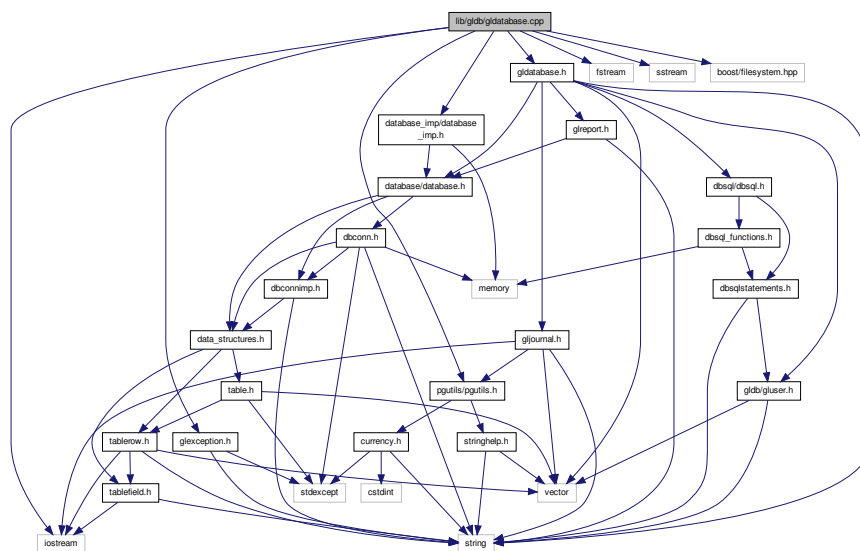
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.30 libgldb/gldatabase.cpp File Reference

Implementation of General Ledger database class.

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <boost/filesystem.hpp>
#include "gldatabase.h"
#include "glexception.h"
#include "database_imp/database_imp.h"
#include "pgutils/pgutils.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" })

10.30.1 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/>

10.30.2 Function Documentation

10.30.2.1 `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

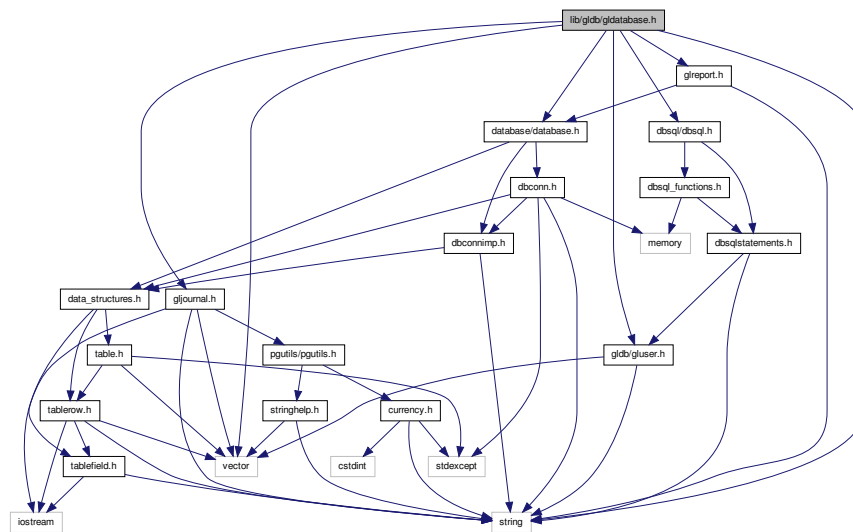
<i>GLDBException</i>	if <i>bs</i> contains any other value.
----------------------	--

10.31 lib/gldb/gldatabase.h File Reference

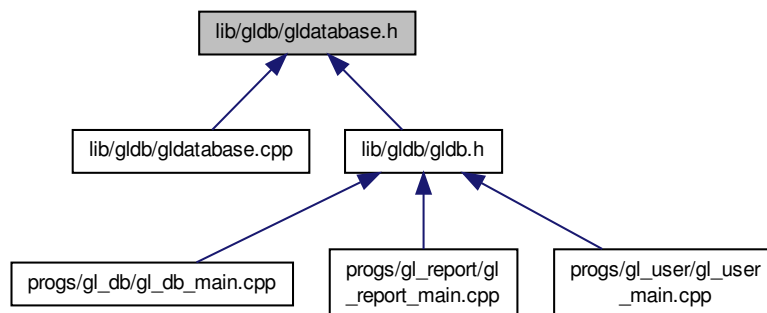
Interface to General Ledger database class.

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

10.31.1 Detailed Description

Interface to General Ledger database class.

Author

Paul Griffiths

Copyright

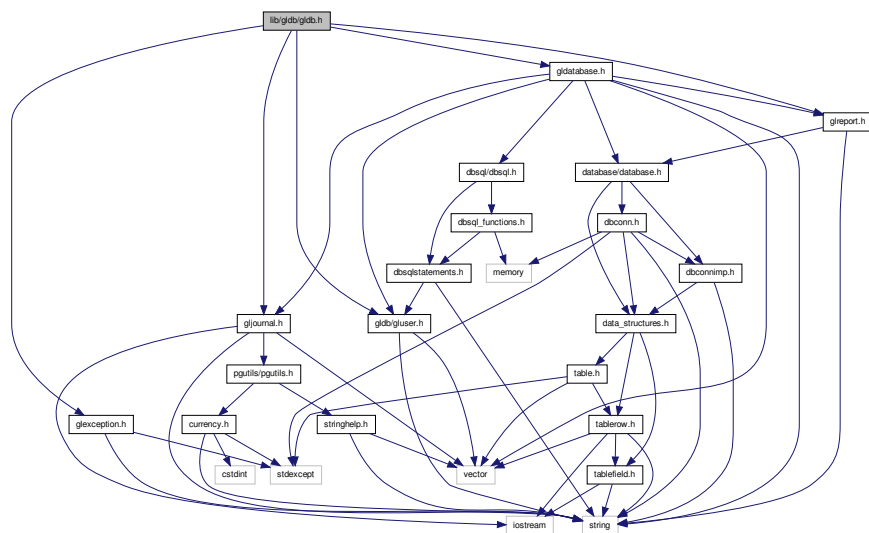
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.32 lib/gldb/gldb.h File Reference

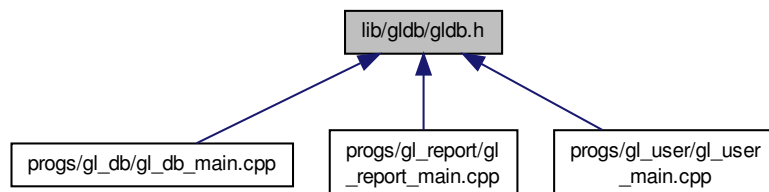
User interface to General Ledger database module.

```
#include "glexception.h"
#include "gldatabase.h"
#include "gluser.h"
#include "glreport.h"
#include "gljournal.h"
```

Include dependency graph for gldb.h:



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



10.32.1 Detailed Description

User interface to General Ledger database module.

Author

Paul Griffiths

Copyright

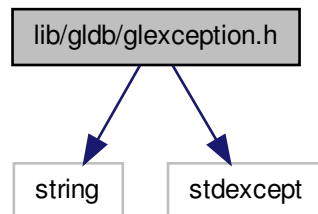
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.33 lib/gldb/glexception.h File Reference

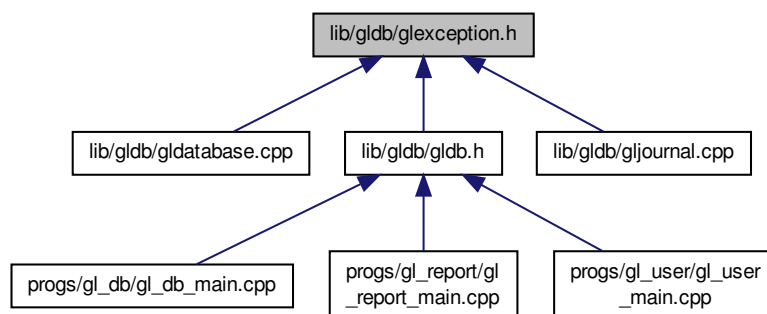
Interface to General Ledger base exception class.

```
#include <string>
#include <stdexcept>
```

Include dependency graph for glexception.h:



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

**Classes**

- class [genleg::GLDBException](#)

Base general ledger database exception class.

10.33.1 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/>

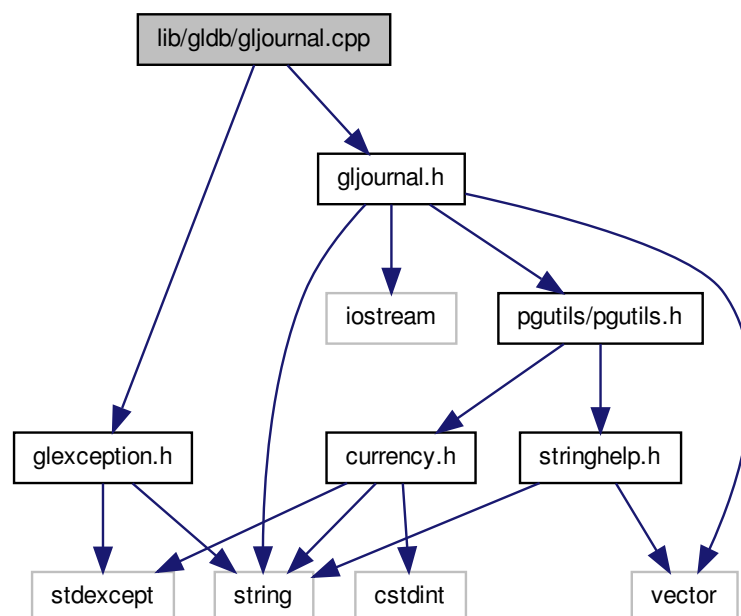
10.34 lib/gldb/gljournal.cpp File Reference

Implementation of journal entry classes.

```
#include "gljournal.h"
```

```
#include "glexception.h"
```

Include dependency graph for gljournal.cpp:



10.34.1 Detailed Description

Implementation of journal entry classes.

Author

Paul Griffiths

Copyright

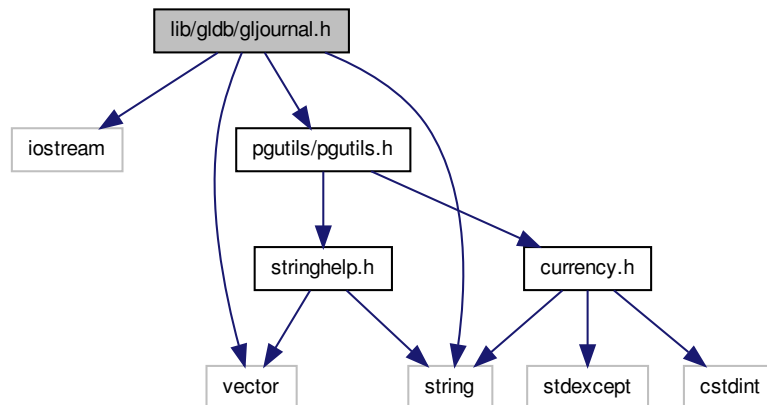
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.35 lib/gldb/gljournal.h File Reference

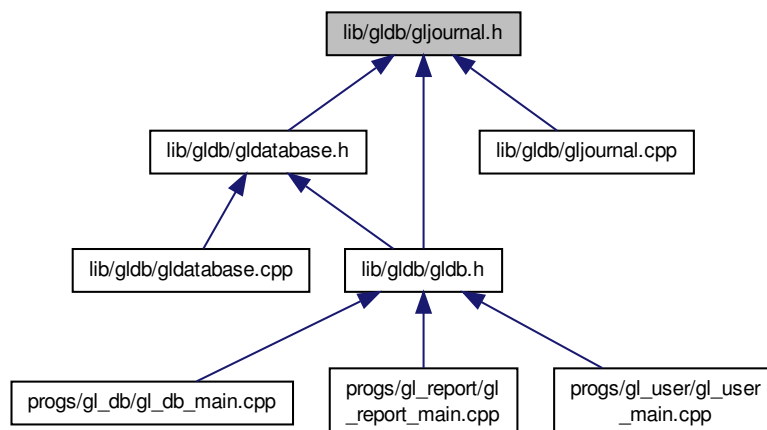
Interface to journal entry classes.

```
#include <iostream>
#include <vector>
#include <string>
#include "pgutils/pgutils.h"
```

Include dependency graph for gljournal.h:



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



Classes

- class [genleg::GLJELine](#)

Journal entry line class.

- class [genleg::GLJournal](#)

Journal entry class.

Functions

- [GLJournal](#) [genleg::journal_from_stream](#) (std::istream &ifs)

Returns a journal entry from a stream in a standard format.

10.35.1 Detailed Description

Interface to journal entry classes.

Author

Paul Griffiths

Copyright

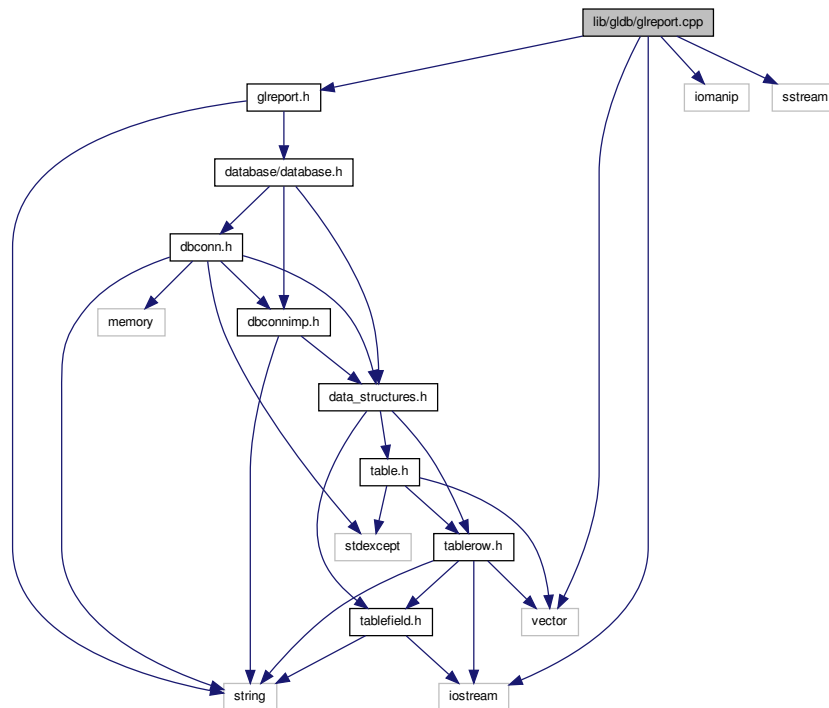
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.36 lib/gldb/glreport.cpp File Reference

Implementation of report class.

```
#include <vector>
#include <iomanip>
#include <iostream>
#include <sstream>
#include "glreport.h"
```

Include dependency graph for glreport.cpp:



Functions

- static std::vector< size_t > [max_column_widths](#) (const [gldb::Table](#) &table)
Calculates the maximum required column widths for a table.
- static void [grow_widths](#) (std::vector< size_t > &widths, const [TableRow](#) &row)
Increments a vector of required column widths.
- static std::string [separator_row](#) (const std::vector< size_t > &widths)
Returns a decorated separator row for a table.
- static std::string [plain_row](#) (const [TableRow](#) &row, const std::vector< size_t > &widths)
Returns a row for a plain report.
- static std::string [decorated_row](#) (const [TableRow](#) &row, const std::vector< size_t > &widths)
Returns a row for a decorated report.

10.36.1 Detailed Description

Implementation of report class.

Author

Paul Griffiths

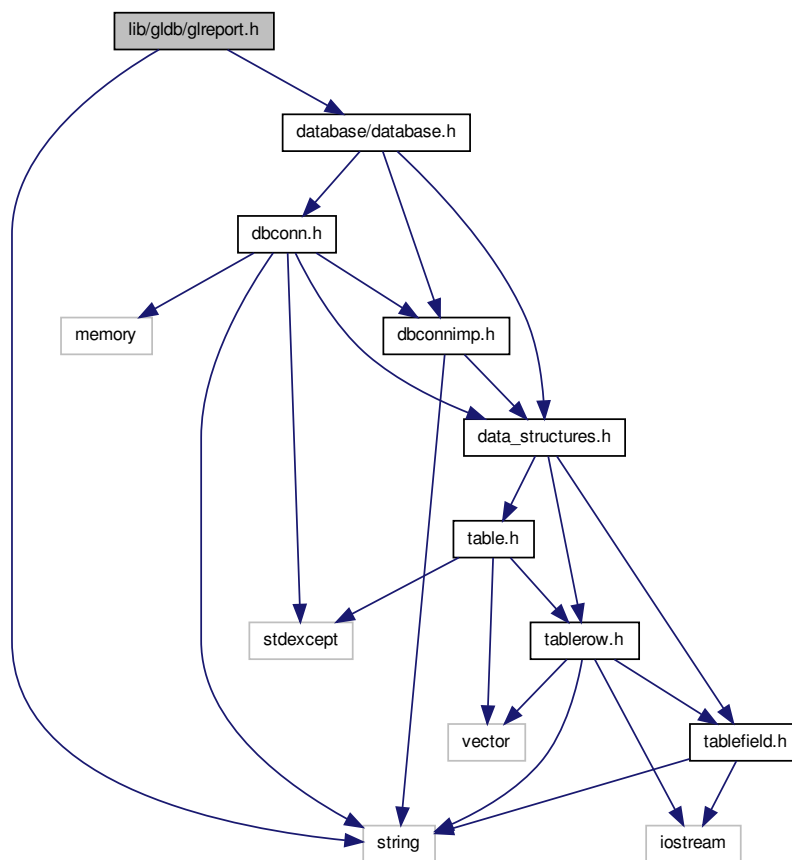
Copyright

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

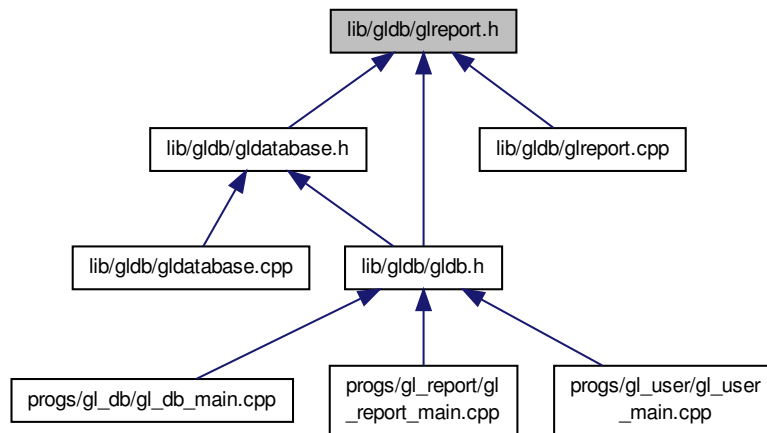
10.37 lib/gldb/glreport.h File Reference

Interface to report class.

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



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



Classes

- class `genleg::GLReport`
General ledger report class.

Functions

- `std::string genleg::plain_report_from_table` (const `gldb::Table` &table)
Creates a plain report from a table.
- `std::string genleg::decorated_report_from_table` (const `gldb::Table` &table)
Creates a decorated report from a table.
- `std::ostream & genleg::operator<<` (std::ostream &out, const `GLReport` &report)
Overridden << operator for printing a report.

10.37.1 Detailed Description

Interface to report class.

Author

Paul Griffiths

Copyright

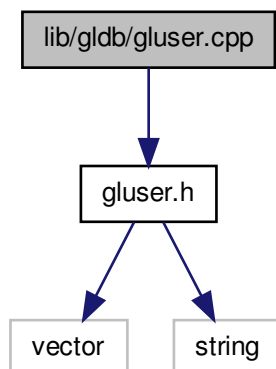
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.38 lib/gldb/gluser.cpp File Reference

Implementation of user class.


```
#include "gluser.h"
```

Include dependency graph for gluser.cpp:



10.38.1 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/>

10.39 lib/gdb/gluser.h File Reference

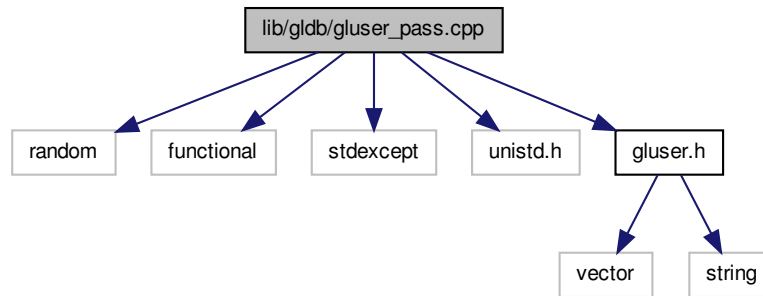
Interface to user class.

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



```
#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()

10.40.1 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/>

10.40.2 Macro Definition Documentation

10.40.2.1 `#define _XOPEN_SOURCE 600`

UNIX feature test macro

10.40.3 Function Documentation

10.40.3.1 `static std::string generate_salt () [static]`

Generates a random two-character salt for `crypt()`

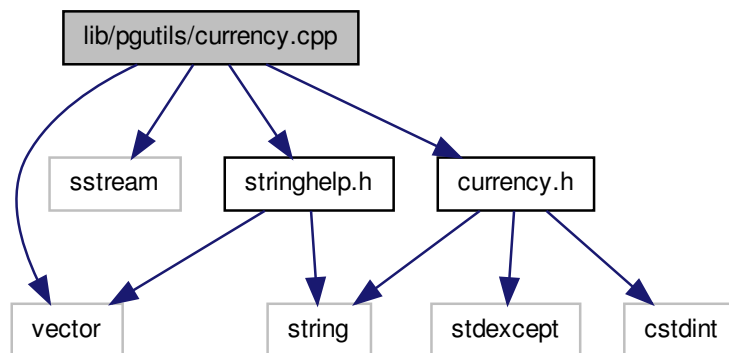
Returns

The two-character salt.

10.41 `lib/pgutils/currency.cpp` File Reference

Implementation of currency amount class.

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



10.41.1 Detailed Description

Implementation of currency amount class.

Author

Paul Griffiths

Copyright

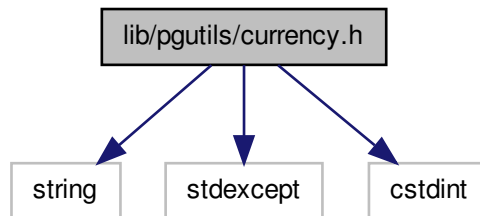
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.42 `lib/pgutils/currency.h` File Reference

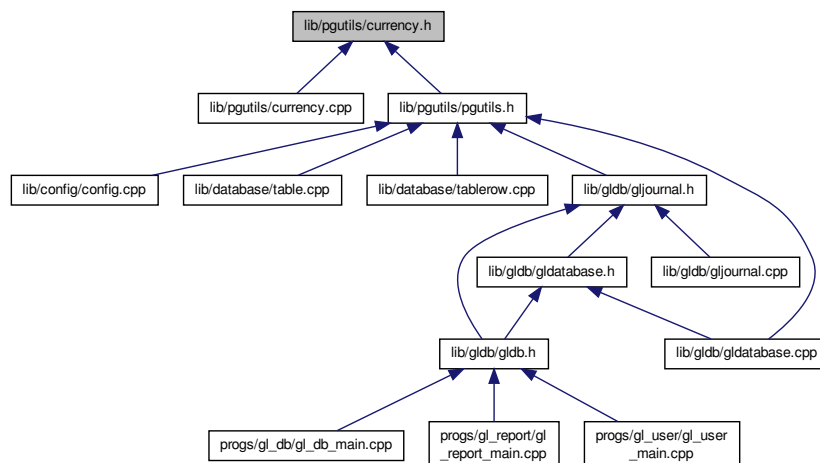
Interface to currency amount class.

```
#include <string>
#include <stdexcept>
#include <cstdint>
```

Include dependency graph for currency.h:



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



Classes

- class `pgutils::CurrencyException`
Base `Currency` exception class.
- class `pgutils::Currency`
`Currency` amount class.

Functions

- `Currency pgutils::operator+ (Currency lhs, const Currency &rhs)`
`Currency` addition operator.
- `Currency pgutils::operator- (Currency lhs, const Currency &rhs)`
`Currency` subtraction operator.

- bool `pgutils::operator==` (const `Currency` &lhs, const `Currency` &rhs)
Currency equality comparison operator.
- bool `pgutils::operator!=` (const `Currency` &lhs, const `Currency` &rhs)
Currency inequality comparison operator.
- bool `pgutils::operator<` (const `Currency` &lhs, const `Currency` &rhs)
Currency less than comparison operator.
- bool `pgutils::operator>` (const `Currency` &lhs, const `Currency` &rhs)
Currency greater than comparison operator.
- bool `pgutils::operator<=` (const `Currency` &lhs, const `Currency` &rhs)
Currency less than or equal to comparison operator.
- bool `pgutils::operator>=` (const `Currency` &lhs, const `Currency` &rhs)
Currency greater than or equal to comparison operator.
- `Currency pgutils::currency_from_string` (const std::string &s)
Creates a currency amount from a string representation.

10.42.1 Detailed Description

Interface to currency amount class.

Author

Paul Griffiths

Copyright

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

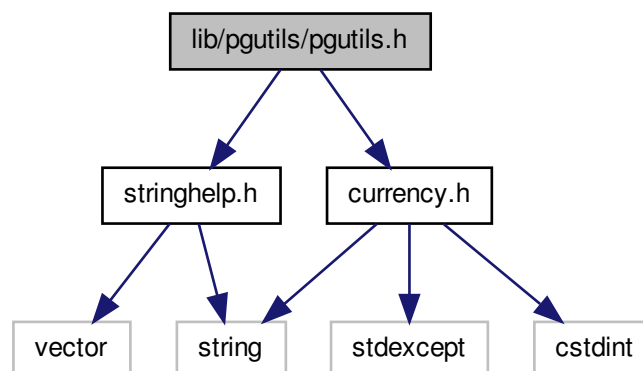
10.43 lib/pgutils/pgutils.h File Reference

Aggregate interface to general utility functions.

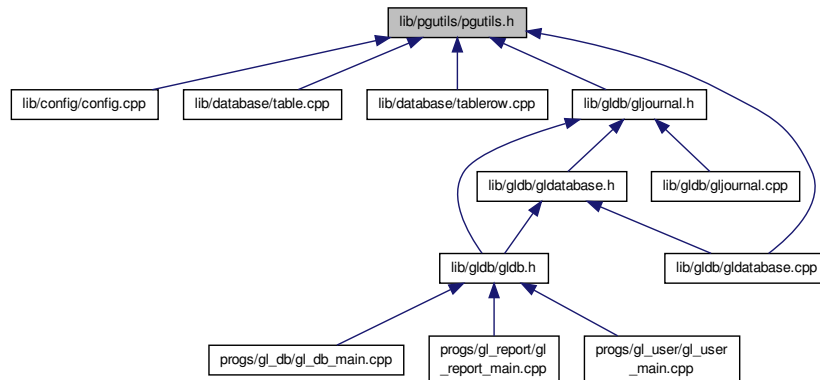
```
#include "stringhelp.h"
```

```
#include "currency.h"
```

Include dependency graph for pgutils.h:



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



10.43.1 Detailed Description

Aggregate interface to general utility functions.

Author

Paul Griffiths

Copyright

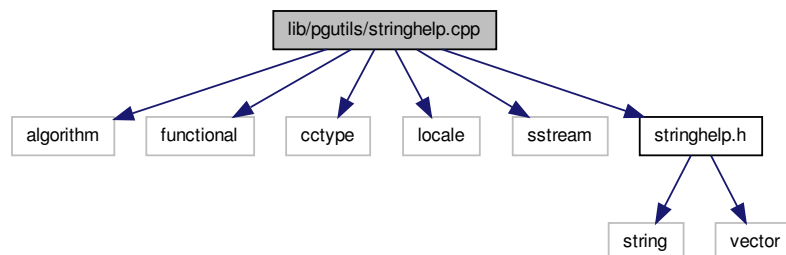
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.44 lib/pgutils/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:



10.44.1 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/>

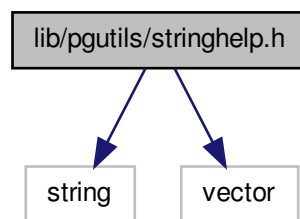
10.45 lib/pgutils/stringhelp.h File Reference

Interface to string helper functions.

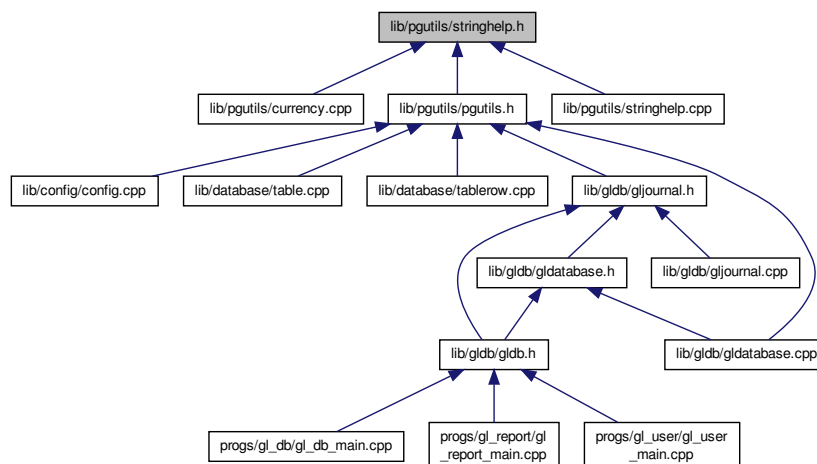
```
#include <string>
```

```
#include <vector>
```

Include dependency graph for stringhelp.h:



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



Functions

- `std::string & pgutils::trim_front` (`std::string &s`)
Trims leading whitespace from a string.
- `std::string & pgutils::trim_back` (`std::string &s`)
Trims trailing whitespace from a string.
- `std::string & pgutils::trim` (`std::string &s`)
Trims leading and trailing whitespace from a string.
- `std::vector< std::string > pgutils::split` (`const std::string &s, const char delim`)
Splits a delimited string into tokens.
- `std::vector< std::string > & pgutils::split` (`std::vector< std::string > &vec, const std::string &s, const char delim`)
Splits a delimited string into tokens.
- `bool pgutils::next_content_line` (`std::istream &if, std::string &s`)
Gets the next content line from a stream.
- `std::vector< std::string > & pgutils::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 > > & pgutils::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 & pgutils::join` (`const std::vector< std::string > &vec, std::string &s, const char delim`)
Joins a vector of strings into a delimited line.
- `bool pgutils::replace` (`std::string &str, const std::string &from, const std::string &to`)
Replaces a substring with another string.

10.45.1 Detailed Description

Interface to string helper functions.

Author

Paul Griffiths

Copyright

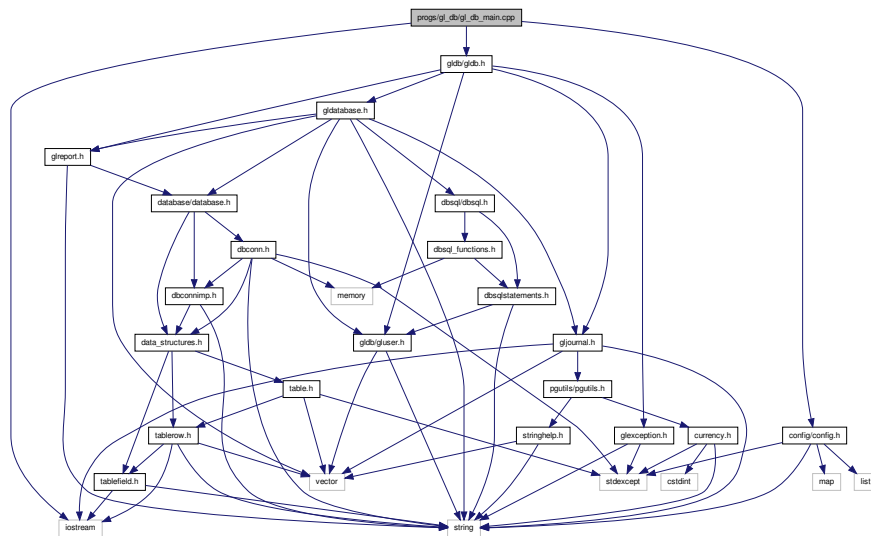
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

10.46 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 * [progname](#) = "gl_db"
Static variable for program name.

10.46.1 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/>

10.47 progs/gl_report/gl_report_main.cpp File Reference

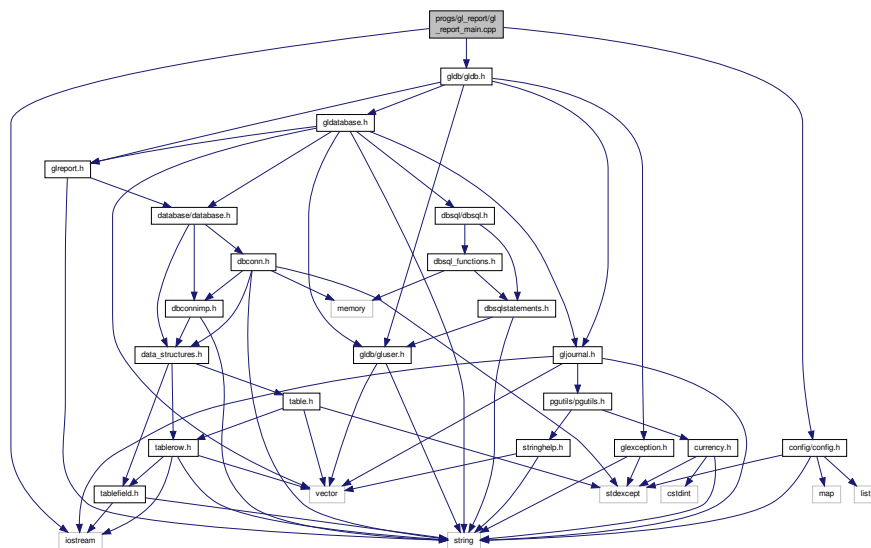
Main functionality for gl_report program.

```
#include <iostream>
```

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

```
#include "config/config.h"
```

Include dependency graph for gl_report_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.

- 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 * `progrname` = "gl_user"
 - Static variable for program name.*

10.48.1 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/>

Index

- ~Config
 - genleg::Config, [37](#)
- ~DBConnDummy
 - gldb::DBConnDummy, [55](#)
- ~DBConnImp
 - gldb::DBConnImp, [57](#)
- ~DBConnMySQL
 - gldb::DBConnMySQL, [60](#)
- ~DBSQLStatements
 - genleg::DBSQLStatements, [65](#)
- ~GLDatabase
 - genleg::GLDatabase, [70](#)
- ~GLReport
 - genleg::GLReport, [79](#)
- ~GLUser
 - genleg::GLUser, [81](#)
- ~MySQLResult
 - gldb::MySQLResult, [85](#)
- ~Table
 - gldb::Table, [89](#)
- ~TableField
 - gldb::TableField, [98](#)
- ~TableRow
 - gldb::TableRow, [107](#)
- _XOPEN_SOURCE
 - config_getopt.cpp, [114](#)
 - gluser_pass.cpp, [161](#)
- account
 - genleg::GLJELine, [75](#)
- add_cmdline_option
 - genleg::Config, [38](#)
- amount
 - genleg::GLJELine, [75](#)
- append_field
 - gldb::TableRow, [107](#), [108](#)
- append_record
 - gldb::Table, [89](#)
- backend
 - genleg::GLDatabase, [70](#)
- begin
 - genleg::GLJournal, [77](#)
 - gldb::Table, [89](#)
 - gldb::TableRow, [108](#)
- boolstring_to_bool
 - gldatabase.cpp, [149](#)
- check_db_parameters
 - Database program., [30](#)
- Reporting program., [32](#)
- User administration program., [34](#)
- check_help_and_version
 - Database program., [30](#)
 - Reporting program., [32](#)
 - User administration program., [35](#)
- check_password
 - genleg::GLUser, [81](#)
- check_user_password
 - User administration program., [35](#)
- Config
 - genleg::Config, [37](#)
- config_getopt.cpp
 - _XOPEN_SOURCE, [114](#)
- ConfigBadConfigFile
 - genleg::ConfigBadConfigFile, [40](#)
- ConfigBadOption
 - genleg::ConfigBadOption, [41](#)
- ConfigCouldNotOpenFile
 - genleg::ConfigCouldNotOpenFile, [43](#)
- ConfigException
 - genleg::ConfigException, [44](#)
- ConfigOptionNotSet
 - genleg::ConfigOptionNotSet, [45](#)
- content_lines
 - General purpose utilities., [24](#)
- create_from_file
 - gldb::Table, [89](#)
- create_structure
 - genleg::GLDatabase, [71](#)
- create_table
 - genleg::DBSQLStatements, [65](#)
- create_user
 - genleg::GLDatabase, [71](#)
- create_view
 - genleg::DBSQLStatements, [65](#)
- Currency
 - pgutils::Currency, [46](#)
- currency_from_string
 - General purpose utilities., [24](#)
- CurrencyException
 - pgutils::CurrencyException, [48](#)
- current_trial_balance_report
 - genleg::GLDatabase, [71](#)
- currentttb
 - genleg::DBSQLStatements, [65](#)
- currentttb_by_entity
 - genleg::DBSQLStatements, [65](#)
- DBConn

- gldb::DBConn, 49, 50
- DBConnCouldNotConnect
 - gldb::DBConnCouldNotConnect, 52
- DBConnCouldNotQuery
 - gldb::DBConnCouldNotQuery, 53
- DBConnDummy
 - gldb::DBConnDummy, 54, 55
- DBConnException
 - gldb::DBConnException, 56
- DBConnImp
 - gldb::DBConnImp, 57
- DBConnMySQL
 - gldb::DBConnMySQL, 59, 60
- DBSQLStatements
 - genleg::DBSQLStatements, 65
- Database interaction module, 16
 - get_connection, 17
 - get_database_type, 17
 - get_field_names, 17
 - get_row, 17
- Database program., 30
 - check_db_parameters, 30
 - check_help_and_version, 30
 - login, 31
 - main, 31
 - set_configuration, 31
- decorated_report_from_table
 - General Ledger database module., 20
- decorated_row
 - General Ledger database module., 21
- destroy_structure
 - genleg::GLDatabase, 71
- drop_table
 - genleg::DBSQLStatements, 66
- drop_view
 - genleg::DBSQLStatements, 66
- enable_user
 - User administration program., 35
- enabled
 - genleg::GLUser, 82
- end
 - genleg::GLJournal, 77
 - gldb::Table, 90
 - gldb::TableRow, 108
- expand
 - pgutils::Currency, 46
- firstname
 - genleg::GLUser, 82
- GLDBException
 - genleg::GLDBException, 74
- GLDatabase
 - genleg::GLDatabase, 70
- GLJournal
 - genleg::GLJournal, 77
- GLReport
 - genleg::GLReport, 79
- GLUser
 - genleg::GLUser, 81
- General Ledger database module., 20
 - decorated_report_from_table, 20
 - decorated_row, 21
 - grow_widths, 21
 - max_column_widths, 21
 - plain_report_from_table, 21
 - plain_row, 21
 - separator_row, 22
- General purpose utilities., 23
 - content_lines, 24
 - currency_from_string, 24
 - join, 24
 - next_content_line, 24
 - operator<, 25
 - operator<=, 26
 - operator>, 26
 - operator>=, 26
 - operator+, 25
 - operator-, 25
 - operator==, 26
 - replace, 27
 - split, 27
 - split_lines, 28
 - trim, 28
 - trim_back, 28
 - trim_front, 28
- generate_salt
 - gluser_pass.cpp, 162
- genleg::Config, 37
 - ~Config, 37
 - add_cmdline_option, 38
 - Config, 37
 - is_set, 38
 - m_opts_set, 39
 - m_opts_supp, 39
 - populate_from_cmdline, 38
 - populate_from_file, 39
- genleg::ConfigBadConfigFile, 39
 - ConfigBadConfigFile, 40
- genleg::ConfigBadOption, 41
 - ConfigBadOption, 41
- genleg::ConfigCouldNotOpenFile, 42
 - ConfigCouldNotOpenFile, 43
- genleg::ConfigException, 43
 - ConfigException, 44
- genleg::ConfigOptionNotSet, 44
 - ConfigOptionNotSet, 45
- genleg::DBSQLDummy, 61
- genleg::DBSQLMySQL, 62
- genleg::DBSQLStatements, 63
 - ~DBSQLStatements, 65
 - create_table, 65
 - create_view, 65
 - currenttb, 65
 - currenttb_by_entity, 65
 - DBSQLStatements, 65

- drop_table, 66
- drop_view, 66
- get_perms, 66
- grant, 66
- listusers, 67
- post_je, 67
- post_je_line, 67
- revoke, 67
- update_user, 68
- user_by_id, 68
- user_by_username, 68
- genleg::GLDBException, 74
 - GLDBException, 74
- genleg::GLDatabase, 68
 - ~GLDatabase, 70
 - backend, 70
 - create_structure, 71
 - create_user, 71
 - current_trial_balance_report, 71
 - destroy_structure, 71
 - GLDatabase, 70
 - get_user_by_id, 71
 - get_user_by_username, 72
 - grant, 72
 - list_users_report, 72
 - load_sample_data, 72
 - m_dbc, 73
 - m_sql, 73
 - m_tables, 74
 - m_views, 74
 - post_journal, 73
 - report, 73
 - revoke, 73
 - update_user, 73
- genleg::GLJELine, 74
 - account, 75
 - amount, 75
 - m_acct, 76
 - m_amount, 76
- genleg::GLJournal, 76
 - begin, 77
 - end, 77
 - GLJournal, 77
 - m_entity, 78
 - m_lines, 78
 - m_memo, 78
 - m_period, 78
 - m_source, 78
 - m_year, 78
- genleg::GLReport, 78
 - ~GLReport, 79
 - GLReport, 79
 - m_report_text, 79
 - operator<<, 79
- genleg::GLUser, 79
 - ~GLUser, 81
 - check_password, 81
 - enabled, 82
 - firstname, 82
 - GLUser, 81
 - id, 82
 - lastname, 82
 - m_enabled, 84
 - m_firstname, 84
 - m_id, 84
 - m_lastname, 84
 - m_pass_hash, 84
 - m_pass_salt, 84
 - m_perms, 84
 - m_username, 84
 - pass_hash, 82
 - pass_salt, 82
 - permissions, 82
 - set_enabled, 83
 - set_firstname, 83
 - set_lastname, 83
 - set_password, 83
 - set_username, 83
 - username, 83
- get_connection
 - Database interaction module, 17
- get_database_type
 - Database interaction module, 17
- get_field
 - gldb::Table, 90
- get_field_names
 - Database interaction module, 17
- get_headers
 - gldb::Table, 90
- get_perms
 - genleg::DBSQLStatements, 66
- get_row
 - Database interaction module, 17
- get_user
 - User administration program., 35
- get_user_by_id
 - genleg::GLDatabase, 71
- get_user_by_username
 - genleg::GLDatabase, 72
- gldatabase.cpp
 - boolstring_to_bool, 149
- gldb::DBConn, 48
 - DBConn, 49, 50
 - last_auto_increment, 50
 - m_imp, 50
 - operator=, 50
 - query, 50
 - select, 50
- gldb::DBConnCouldNotConnect, 51
 - DBConnCouldNotConnect, 52
- gldb::DBConnCouldNotQuery, 52
 - DBConnCouldNotQuery, 53
- gldb::DBConnDummy, 53
 - ~DBConnDummy, 55
 - DBConnDummy, 54, 55
 - operator=, 55

- query, 55
 - select, 55
- gldb::DBConnException, 56
 - DBConnException, 56
- gldb::DBConnImp, 56
 - ~DBConnImp, 57
 - DBConnImp, 57
 - last_auto_increment, 57
 - query, 57
 - select, 58
- gldb::DBConnMySQL, 58
 - ~DBConnMySQL, 60
 - DBConnMySQL, 59, 60
 - last_auto_increment, 60
 - m_conn, 61
 - m_mtx, 61
 - operator=, 60
 - query, 60
 - select, 61
- gldb::MySQLResult, 84
 - ~MySQLResult, 85
 - m_num_fields, 86
 - m_result, 86
 - MySQLResult, 85
 - num_fields, 85
 - operator=, 86
 - result, 86
- gldb::Table, 86
 - ~Table, 89
 - append_record, 89
 - begin, 89
 - create_from_file, 89
 - end, 90
 - get_field, 90
 - get_headers, 90
 - insert_query, 91
 - m_headers, 92
 - m_quoted, 92
 - m_records, 92
 - num_fields, 91
 - num_records, 91
 - operator=, 91
 - set_quoted, 92
 - Table, 88, 89
- gldb::TableBadInputFile, 93
 - TableBadInputFile, 93
- gldb::TableCouldNotOpenInputFile, 94
 - TableCouldNotOpenInputFile, 95
- gldb::TableException, 95
 - TableException, 96
- gldb::TableField, 96
 - ~TableField, 98
 - length, 98
 - m_data, 101
 - operator std::string, 98
 - operator<<, 101
 - operator+=, 98, 99
 - operator=, 99, 100
 - TableField, 97, 98
- gldb::TableMismatchedRecordLength, 101
 - TableMismatchedRecordLength, 102
- gldb::TableNoSuchField, 102
 - TableNoSuchField, 103
- gldb::TableNoSuchRecord, 104
 - TableNoSuchRecord, 104
- gldb::TableRow, 105
 - ~TableRow, 107
 - append_field, 107, 108
 - begin, 108
 - end, 108
 - m_fields, 110
 - operator=, 109
 - print, 109
 - record_string, 110
 - size, 110
 - TableRow, 106, 107
- gluser_pass.cpp
 - _XOPEN_SOURCE, 161
 - generate_salt, 162
- grant
 - genleg::DBSQLStatements, 66
 - genleg::GLDatabase, 72
- grow_widths
 - General Ledger database module., 21
- id
 - genleg::GLUser, 82
- insert_query
 - gldb::Table, 91
- is_set
 - genleg::Config, 38
- join
 - General purpose utilities., 24
- last_auto_increment
 - gldb::DBConn, 50
 - gldb::DBConnImp, 57
 - gldb::DBConnMySQL, 60
- lastname
 - genleg::GLUser, 82
- length
 - gldb::TableField, 98
- lib/config/config.cpp, 111
- lib/config/config.h, 112
- lib/config/config_getopt.cpp, 113
- lib/database/data_structures.h, 114
- lib/database/database.h, 115
- lib/database/dbconn.cpp, 117
- lib/database/dbconn.h, 118
- lib/database/dbconnimp.h, 119
- lib/database/table.cpp, 121
- lib/database/table.h, 122
- lib/database/tablefield.cpp, 124
- lib/database/tablefield.h, 124
- lib/database/ablerow.cpp, 126
- lib/database/ablerow.h, 127

- lib/database_imp/database_imp.h, 128
- lib/database_imp/dummy/dbconn_dummy_imp.cpp, 130
- lib/database_imp/dummy/dbconn_dummy_imp.h, 131
- lib/database_imp/mysql/dbconn_mysql_functions.cpp, 133
- lib/database_imp/mysql/dbconn_mysql_imp.cpp, 134
- lib/database_imp/mysql/dbconn_mysql_imp.h, 136
- lib/database_imp/mysql/dbconn_mysql_result.cpp, 137
- lib/database_imp/mysql/dbconn_mysql_result.h, 138
- lib/dbsql/dbsql.h, 139
- lib/dbsql/dbsql_dummy.h, 140
- lib/dbsql/dbsql_functions.h, 142
- lib/dbsql/dbsql_implementations.h, 143
- lib/dbsql/dbsql_mysql.h, 145
- lib/dbsql/dbsqlstatements.cpp, 146
- lib/dbsql/dbsqlstatements.h, 147
- lib/gldb/gldatabase.cpp, 148
- lib/gldb/gldatabase.h, 149
- lib/gldb/gldb.h, 151
- lib/gldb/glexception.h, 152
- lib/gldb/gljournal.cpp, 153
- lib/gldb/gljournal.h, 154
- lib/gldb/glreport.cpp, 155
- lib/gldb/glreport.h, 157
- lib/gldb/gluser.cpp, 158
- lib/gldb/gluser.h, 159
- lib/gldb/gluser_pass.cpp, 160
- lib/pgutils/currency.cpp, 162
- lib/pgutils/currency.h, 162
- lib/pgutils/pgutils.h, 164
- lib/pgutils/stringhelp.cpp, 165
- lib/pgutils/stringhelp.h, 166
- list_users_report
 - genleg::GLDatabase, 72
- listusers
 - genleg::DBSQLStatements, 67
- load_sample_data
 - genleg::GLDatabase, 72
- login
 - Database program., 31
 - Reporting program., 33
 - User administration program., 35
- m_acct
 - genleg::GLJELine, 76
- m_amount
 - genleg::GLJELine, 76
- m_conn
 - gldb::DBConnMySQL, 61
- m_data
 - gldb::TableField, 101
- m_dbc
 - genleg::GLDatabase, 73
- m_enabled
 - genleg::GLUser, 84
- m_entity
 - genleg::GLJournal, 78
- m_fields
 - gldb::TableRow, 110
- m_firstname
 - genleg::GLUser, 84
- m_frac
 - pgutils::Currency, 48
- m_headers
 - gldb::Table, 92
- m_id
 - genleg::GLUser, 84
- m_imp
 - gldb::DBConn, 50
- m_int
 - pgutils::Currency, 48
- m_lastname
 - genleg::GLUser, 84
- m_lines
 - genleg::GLJournal, 78
- m_memo
 - genleg::GLJournal, 78
- m_mtx
 - gldb::DBConnMySQL, 61
- m_num_fields
 - gldb::MySQLResult, 86
- m_opts_set
 - genleg::Config, 39
- m_opts_supp
 - genleg::Config, 39
- m_pass_hash
 - genleg::GLUser, 84
- m_pass_salt
 - genleg::GLUser, 84
- m_period
 - genleg::GLJournal, 78
- m_perms
 - genleg::GLUser, 84
- m_quoted
 - gldb::Table, 92
- m_records
 - gldb::Table, 92
- m_report_text
 - genleg::GLReport, 79
- m_result
 - gldb::MySQLResult, 86
- m_source
 - genleg::GLJournal, 78
- m_sql
 - genleg::GLDatabase, 73
- m_tables
 - genleg::GLDatabase, 74
- m_username
 - genleg::GLUser, 84
- m_views
 - genleg::GLDatabase, 74
- m_year
 - genleg::GLJournal, 78
- main
 - Database program., 31
 - Reporting program., 33
 - User administration program., 36

- max_column_widths
 - General Ledger database module., 21
- MySQLResult
 - gldb::MySQLResult, 85
- next_content_line
 - General purpose utilities., 24
- num_fields
 - gldb::MySQLResult, 85
 - gldb::Table, 91
- num_records
 - gldb::Table, 91
- operator std::string
 - gldb::TableField, 98
- operator<
 - General purpose utilities., 25
 - pgutils::Currency, 47
- operator<<
 - genleg::GLReport, 79
 - gldb::TableField, 101
- operator<=
 - General purpose utilities., 26
- operator>
 - General purpose utilities., 26
- operator>=
 - General purpose utilities., 26
- operator+
 - General purpose utilities., 25
 - pgutils::Currency, 47
- operator+=
 - gldb::TableField, 98, 99
 - pgutils::Currency, 46
- operator-
 - General purpose utilities., 25
 - pgutils::Currency, 46
- operator-=
 - pgutils::Currency, 46
- operator=
 - gldb::DBConn, 50
 - gldb::DBConnDummy, 55
 - gldb::DBConnMySQL, 60
 - gldb::MySQLResult, 86
 - gldb::Table, 91
 - gldb::TableField, 99, 100
 - gldb::TableRow, 109
- operator==
 - General purpose utilities., 26
 - pgutils::Currency, 47
- pass_hash
 - genleg::GLUser, 82
- pass_salt
 - genleg::GLUser, 82
- permissions
 - genleg::GLUser, 82
- pgutils::Currency, 45
 - Currency, 46
 - expand, 46
 - m_frac, 48
 - m_int, 48
 - operator<, 47
 - operator+, 47
 - operator+=", 46
 - operator-, 46
 - operator-=, 46
 - operator==, 47
- pgutils::CurrencyException, 48
 - CurrencyException, 48
- plain_report_from_table
 - General Ledger database module., 21
- plain_row
 - General Ledger database module., 21
- populate_from_cmdline
 - genleg::Config, 38
- populate_from_file
 - genleg::Config, 39
- post_je
 - genleg::DBSQLStatements, 67
- post_je_line
 - genleg::DBSQLStatements, 67
- post_journal
 - genleg::GLDatabase, 73
- print
 - gldb::TableRow, 109
- Program configuration module, 15
- progs/gl_db/gl_db_main.cpp, 167
- progs/gl_report/gl_report_main.cpp, 169
- progs/gl_user/gl_user_main.cpp, 170
- query
 - gldb::DBConn, 50
 - gldb::DBConnDummy, 55
 - gldb::DBConnImp, 57
 - gldb::DBConnMySQL, 60
- record_string
 - gldb::TableRow, 110
- replace
 - General purpose utilities., 27
- report
 - genleg::GLDatabase, 73
- Reporting program., 32
 - check_db_parameters, 32
 - check_help_and_version, 32
 - login, 33
 - main, 33
 - set_configuration, 33
- result
 - gldb::MySQLResult, 86
- revoke
 - genleg::DBSQLStatements, 67
 - genleg::GLDatabase, 73
- SQL statements module, 19
- select
 - gldb::DBConn, 50
 - gldb::DBConnDummy, 55

- gldb::DBConnImp, 58
 - gldb::DBConnMySQL, 61
- separator_row
 - General Ledger database module., 22
- set_configuration
 - Database program., 31
 - Reporting program., 33
 - User administration program., 36
- set_enabled
 - genleg::GLUser, 83
- set_firstname
 - genleg::GLUser, 83
- set_lastname
 - genleg::GLUser, 83
- set_password
 - genleg::GLUser, 83
- set_quoted
 - gldb::Table, 92
- set_user_password
 - User administration program., 36
- set_username
 - genleg::GLUser, 83
- show_user_details
 - User administration program., 36
- size
 - gldb::TableRow, 110
- split
 - General purpose utilities., 27
- split_lines
 - General purpose utilities., 28
- Table
 - gldb::Table, 88, 89
- TableBadInputFile
 - gldb::TableBadInputFile, 93
- TableCouldNotOpenInputFile
 - gldb::TableCouldNotOpenInputFile, 95
- TableException
 - gldb::TableException, 96
- TableField
 - gldb::TableField, 97, 98
- TableMismatchedRecordLength
 - gldb::TableMismatchedRecordLength, 102
- TableNoSuchField
 - gldb::TableNoSuchField, 103
- TableNoSuchRecord
 - gldb::TableNoSuchRecord, 104
- TableRow
 - gldb::TableRow, 106, 107
- trim
 - General purpose utilities., 28
- trim_back
 - General purpose utilities., 28
- trim_front
 - General purpose utilities., 28
- update_user
 - genleg::DBSQLStatements, 68
 - genleg::GLDatabase, 73
- User administration program., 34
 - check_db_parameters, 34
 - check_help_and_version, 35
 - check_user_password, 35
 - enable_user, 35
 - get_user, 35
 - login, 35
 - main, 36
 - set_configuration, 36
 - set_user_password, 36
 - show_user_details, 36
- user_by_id
 - genleg::DBSQLStatements, 68
- user_by_username
 - genleg::DBSQLStatements, 68
- username
 - genleg::GLUser, 83